Piecewise Notes

Ally Macdonald

Contents

1 Introduction

These notes serve as a single source of knowledge in which those reading should ideally be able to garner a deeper understanding of the thoughts and ideas about piecewise objects I have had and continue to have.

To consider something piecewise is to enumerate something under differing conditions. For example, one might consider the function $|x|$ piecewise, namely $|x| = x$ when $x \geq 0$ or $|x| = -x$ when $x \leq 0$.

An object which is piecewise is something which contains a set of pieces, which themselves contain values and conditions, under which those values are taken for the overall object. Continuing with the example of $|x|$, we have the piece value x for when $x \geq 0$, the condition, (which forms a piece), and $-x$ for when $x \leq 0$.

We shall develop ideas such as above more explicitly throughout these notes. These ideas often intersect with other areas of maths, which one might be familiar with — but if not, don't panic; such ideas are not strictly foundational to the presented notes.

Furthermore, these notes focus on construction; rather than evaluating existing concepts or formulas, we focus on deriving existing or new tools, ideas and formulas. For you, reader, this must be a process you should become familiar with, and rather than just reading these notes, attempt to follow along by hand, and construct your own objects using the ideas presented here. We also stress that with ideas that intersect with more mainstream mathematics, that existing concepts be used to evaluate the validity of the constructions.

Finally, if you have trouble understanding some of the ideas presented here, you should consider taking a look at the blog posts on `https://piecewise.org` or contacting myself at ally@piecewise.org. The things you'll see here are the culmination of many hundreds of hours scribbling, doodling and refining thoughts over the course of several years. More importantly than the fact I am myself still learning the fundamentals of mathematics at a higher level, is that you understand that trial and lots of error forms the majority of this work.

Thank you in advance for reading.

2 Notation

2.1 Single variable piecewise functions

A piecewise *function* is a function defined over several pieces.

Let us consider some function $f : \mathbb{R} \to \mathbb{R}$ and some intervals $D_1, D_2, \ldots, D_n \subseteq \mathbb{R}$ such that $D_1 \cap D_2 \cap \cdots \cap D_n = \varnothing$. That is, we have some number real intervals which do not overlap. Suppose we have some functions $f_1 : D_1 \to \mathbb{R}$ and so on. We describe this function using the following notation:

$$f(x) = \begin{cases} f_1(x) & x \in D_1 \\ f_2(x) & x \in D_2 \\ \vdots & \vdots \\ f_n(x) & x \in D_n \end{cases}$$

This same function could instead be written as the following:

$$\begin{gathered} f : D_1 \to \mathbb{R}, f(x) = f_1(x), \\ f : D_2 \to \mathbb{R}, f(x) = f_2(x), \\ \vdots \\ f : D_n \to \mathbb{R}, f(x) = f_n(x) \end{gathered}$$

This essentially describes the same function over different domains. Alternatively, one might think about it as different functions under the same label, f.

Reading this function more verbosely, we say that if x is in the domain D_1 and so on, then we might 'choose' to have $f(x) = f_1(x)$ as per our definition.

2.1. Let us define the absolute value function, $|x|$, as the following:

$$|x| = \begin{cases} x & x \geq 0 \\ -x & x \leq 0 \end{cases}$$

To evaluate, for example, $|-3|$, we use the definition:

$$|-3| = \begin{cases} -3 & -3 \geq 0 \\ 3 & -3 \leq 0 \end{cases}$$

Since the condition in the second piece (or case) evaluates to true, we use the value in that piece. That is, since $-3 \leq 0$ is true, $|-3| = 3$.

2.2 Piecewise functions

Piecewise functions needn't be restricted to a single variable. More generally, piecewise functions needn't be restricted to any number of conditions or pieces.

The perfect example, as we'll see later, is the floor function, which has an infinite number of pieces. Similarly, the bivariate function describing a square when it intersects a certain plane has 4 pieces.

In general, we can notate such a more general piecewise function like the following:

$$f(x) = \begin{cases} f_1(x) & C_{1,1} & C_{1,2} & \dots & C_{1,m} \\ f_2(x) & C_{2,1} & C_{2,2} & \dots & C_{2,m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_n(x) & C_{n,1} & C_{n,2} & \dots & C_{n,m} \end{cases}$$

Where $C_{p,q}$ is some condition which evaluates to true or false for some inputs (e.g. a predicate). We interpret the string of conditions in each piece, such as $\{C_{1,1} \quad C_{1,2} \quad \dots \quad C_{1,m}\}$, as equivalent to the condition $C_{1,1} \wedge C_{1,2} \wedge \cdots \wedge C_{1,m}$.

Such conditions could be anything, and per the section above, we could have $C_{p,q}$ denote the predicate $x \in D_{p,q}$ for some set $D_{p,q}$.

Importantly, we note in general this piecewise object is written in a shortened form, without commas or English terms such as 'and' or 'or', or their respective logical operators. This is a conventional decision in these notes given the number of piecewise objects written. For example, the following is identical to the above:

$$f(x) = \begin{cases} f_1(x), & \text{if } C_{1,1} \wedge C_{1,2} \wedge \cdots \wedge C_{1,m} \\ f_2(x), & \text{if } C_{2,1} \wedge C_{2,2} \wedge \cdots \wedge C_{2,m} \\ \vdots & \vdots \\ f_n(x), & \text{if } C_{n,1} \wedge C_{n,2} \wedge \cdots \wedge C_{n,m} \end{cases}$$

And of course we might package each piece's conditions together, effectively making one condition per piece. Though, in practice, this doesn't help very much at all (this is because, as we will later see, considering each condition individually yields results which affect the way we interpolate functions and other manipulation of piecewise objects).

2.3 Generalised piecewise object

Finally, we might consider a set-like generalised form of a piecewise object. Let us denote the following:

$$\phi = \{\varphi_i, \quad C_i \mid i \in I\}$$

Where ϕ describes the piecewise object, and for each $i \in I$ (the iterator), $\phi = \varphi_i$ for when C_i is true.

2.2. Suppose we have a set $I = \{1, 2, \ldots, n\}$ and a piecewise function f such that $f = f_i$ when C_i is true.

We can express this function as:

$$f = \begin{cases} f_1 & C_1 \\ f_2 & C_2 \\ \vdots & \vdots \\ f_n & C_n \end{cases}$$

Alternatively, we might express it using the generalised notation:

$$f = \{f_i, \quad C_i \mid i \in \{1, 2, \ldots, n\}\}$$

2.3. We will see the floor function later on where it will be described more deeply, but we can introduce it like so using our notation:

$$\lfloor x \rfloor = \{n, \quad x \in [n, n+1) \mid n \in \mathbb{Z}\}$$

To evaluate this function at $x = 3.5$ consider that for all $n \in \mathbb{Z}$ there exists only one such interval $N = [n, n+1)$ such that $3.5 \in N$. This interval is $[3, 4)$, i.e. $n = 3$; the corresponding piece value is 3 and so $\lfloor 3.5 \rfloor = 3$.

If a single iterator (e.g. $i \in I$) isn't sufficient to describe the piecewise object we desire, we might consider using several; $i \in I$, $j \in J$ for example). Alternatively, we might consider joining two piecewise objects together using $\cup$:

$$\varphi = \varphi_1 \cup \varphi_2 = \{\varphi_{1,i}, \quad A_i \mid i \in I\} \cup \{\varphi_{2,j}, \quad B_j \mid j \in J\}$$

This can be interpreted similarly to before, though we now have that $\varphi = \varphi_{1,i}$ if A_i is true for some i, or $\varphi = \varphi_{2,j}$ is B_j is true for some j.

Notably, this notation can be used to express individual pieces explicitly in a piecewise object rather than collectively. If such a piece isn't dependent on the iterator, we can ignore that part entirely (keeping in mind we might be able to rewrite it using an iterator later on).

2.4. Recall the absolute value function; see Example 2.1.

We can then express $|x|$ as:

$$|x| = \{x, \quad x \geq 0\} \cup \{-x, \quad x \leq 0\}$$

3 Algebra on Piecewise Objects

3.1 Piece association

Operations on piecewise objects are fairly straightforward and behave as any other object in the context you're working in (whether this be with the real numbers, complex numbers, certain algebras, spaces, and so on).

What this section aims to do is not to teach you directly how to perform exactly the operations you want to on each piecewise object, but instead provide a basis for which you can base your ideas: no piece of a piecewise object is independent of another. Equivalently, each piece is dependent on each other piece. There is an intuitive reason for this: you are evaluating an object not conditionally, but in full generality; each piece exists, in some sense, 'simultaneously'.

It is here we might recognise that this idea could fairly easily lead into combinatorics; an area of maths which deals heavily with the enumeration and construction of such objects.

3.2 Equality property

3.1. Consider the following piecewise object:

$$\phi = \{\varphi_i, \quad C_i \mid i \in I\}$$

Then if for all $i, j \in I$ we have that $\varphi_i = \varphi_j$ (that is, all piece values are equal for when ϕ is defined), then $\phi = \varphi_i$ for any $i \in I$.

For example, consider the function $f : (-\infty, 0] \to \mathbb{R}$:

$$f(x) = \begin{cases} 0 & x \geq 0 \\ 0 & x < 0 \end{cases}$$

We have that $f(x) = 0$. This is because everywhere in $\mathrm{dom}(f)$, or both when $x \geq 0$ and $x < 0$, we have that $f(x) = 0$.

3.3 Piece and condition equivalence

Arguably one of the most fundamental property of piecewise objects is the relationship between each piece's conditions and its respective value. That is, the values for which a piecewise object takes is dependent on their respective conditions, and so can be treated explicitly as such.

3.2. Consider the following piecewise function:

$$f(x) = \{f_i(x), \quad C_i \mid i \in I\}$$

For all $i \in I$, let us define a substitution $x \sim y_i$, where $\sim$ represents equality under the condition C_i. Then we might consider writing $f(x)$ as:

$$f(x) = \{f_i(y_i), \quad C_i \mid i \in I\}$$

And $f(x)$ may still remain a non-constant function, but in this way we've given it another representation. For some C_i we might just have $x \sim x$ (and is effectively no substitution at all). This substitution can also go the other way, i.e. $y_i \sim x$.

3.3. Consider the following piecewise object:

$$\phi = \{\varphi_i, \quad C_i \mid i \in I\}$$

For all $i \in I$, suppose that $C_i \leftrightarrow D_i$ in the context of ϕ (i.e. if a function, within its domain, etc.). Then we can rewrite this object as:

$$\phi = \{\varphi_i, \quad D_i \mid i \in I\}$$

That is, we've substituted the set of our conditions for another set of conditions (and again, we may not have substituted all of them). For example, we know that $x = 5$ when $(x-5)^2 \leq 0$, and so these two conditions are interchangeable.

This idea becomes far more important when working using other functions, such as max, min and $|x|$ (which will be covered later), although it can still be used in other contexts. In essence, for certain classes of piecewise functions, we can build off existing functions to construct not only transformations, but completely new functions and representations.

3.1. As an example, let us consider the function $f : (-6, 6) \to \mathbb{R}$:

$$f(x) = \begin{cases} x & x > 5 \\ 5 & x = 5 \\ x & x < 5 \end{cases}$$

Consider the piece for which $x = 5$: note that the piece value is equal to 5. Since $x = 5$ we can replace 5 with x to get:

$$f(x) = \begin{cases} x & x > 5 \\ x & x = 5 \\ x & x < 5 \end{cases}$$

And by the equality property, we know that this function is equivalent to $f(x) = x$.

Finally, depending on the context of the problem, function or object, we can rewrite conditions in some way to make it more straightforward for us to understand. This becomes relevant in the context of function domains.

3.2. Let us consider the function from Example 3.1.

The following two representations are equivalent:

$$f(x) = \begin{cases} x & x > 5 \\ 5 & x = 5 \\ x & x < 5 \end{cases}$$

$$f(x) = \begin{cases} x & x \in (5, 6) \\ 5 & x = 5 \\ x & x \in (-6, 5) \end{cases}$$

The reason for this is that for all $x \in (-6, 5)$ we have that $x < 5$, despite the contrary not being always true. The reason the contrary needn't always be true for all $\mathbb{R}$ is because our function f is defined on the interval $(-6, 6)$. This same argument can also be applied to $x \in (5, 6) \implies x > 5$.

A strong note on functions which are not well-defined: Piecewise objects can easily define not well-defined functions. In these cases, care must be taken when working with pieces in general, values or conditions.

3.4. A piecewise function is only well-defined if all of its piece values denote well-defined functions, and in places where two conditions in separate pieces are true, their respective values must be equal.

Formally, given the piecewise object $\phi = \{\varphi_i, \quad C_i \mid i \in I\}$ if there exists $i \neq j \in I$ such that $C_i \wedge C_j$ is true, then ϕ is well-defined iff $\varphi_i = \varphi_j$ and each φ_k for $k \in I$ is well-defined.

For example, the following is not well-defined:

$$f(x) = \begin{cases} 1 & x \in \mathbb{Q} \\ 0 & x \in \mathbb{R} \setminus \mathbb{Z} \end{cases}$$

This is because the latter piece is 0 for all real numbers that aren't integers, and the first piece is 1 for all rational numbers; these include non-integers. There are a few ways we could change this function to be well-defined:

$$f(x) = \begin{cases} 1 & x \in \mathbb{Q} \\ 0 & x \in \mathbb{R} \setminus \mathbb{Q} \end{cases}$$

This is known as the Dirichlet function; it's the indicator function of the rationals (which we'll cover later on). Alternatively:

$$f(x) = \begin{cases} 0 & x \in \mathbb{Q} \\ 0 & x \in \mathbb{R} \setminus \mathbb{Z} \end{cases}$$

Which would simply make the function 0 for all $\mathbb{R}$.

3.4 (De)nesting pieces; logical 'and'

The third of several special operations on piecewise objects involves the nesting and denesting of cases; essentially, we decouple the conditions in each piece from one another in order to group and subsequently simplify the piece values.

3.3. Consider the following piecewise object:

$$\phi = \begin{cases} \varphi_1 & C_1 \\ \varphi_2 & C_2 \\ \vdots & \vdots \\ \varphi_n & C_n \end{cases}$$

Suppose that we have

$$\varphi_1 = \begin{cases} \mu_1 & D_1 \\ \mu_2 & D_2 \end{cases}$$

Then we can rewrite ϕ as:

$$\phi = \begin{cases} \begin{cases} \mu_1 & D_1 \\ \mu_2 & D_2 \end{cases} & C_1 \\ \varphi_2 & C_2 \\ \vdots & \vdots \\ \varphi_n & C_n \end{cases}$$

Suppose that $D_1 \wedge C_1$ is true: then it stands to reason that $\phi = \mu_1$. Likewise, if $D_2 \wedge C_1$ then $\phi = \mu_2$. The behaviour is as normal for each other piece. This also means ϕ can be represented as:

$$\phi = \begin{cases} \mu_1 & C_1 & D_1 \\ \mu_2 & C_1 & D_2 \\ \varphi_2 & C_2 & \\ \vdots & \vdots & \\ \varphi_n & C_n & \end{cases}$$

And so we might go back and forth between these two forms to represent the same object, keeping in mind each column of the piecewise object, other than the first, represents the conditions under which that piece value is taken (the logical 'and').

In general, if we have common, and multiple, conditions for pieces, we should be able to nest, or denest, piecewise objects. This often helps simplify piecewise problems in multiple variables, or single variables with intervals, etc.

3.5. Given the following piecewise object:

$$\phi = \{\varphi_i, \quad C_i \mid i \in I\}$$

If we have that $C_i \leftrightarrow A_i \wedge B_i$ for some conditions A_i, B_i, then

$$\phi = \{\varphi_i, \quad A_i \wedge B_i \mid i \in I\}$$

This is a semi-obvious fact, but becomes useful when coupled with the grouping/nesting of piecewise objects via their piece's conditions. This was also touched on in Chapter 1.

3.5 Combining, splitting piece conditions; logical 'or'

Finally, we'll look at explicit logical 'or' in piece conditions at a basic level and in full generality. This is just as important as the previous section, particularly for grouping.

3.6. Given the following piecewise object:

$$\phi = \{\varphi_i, \quad A_i \vee B_i \mid i \in I\}$$

For some conditions A_i, B_i, we can rewrite this as the following:

$$\phi = \{\varphi_i, \quad A_i \mid i \in I\} \cup \{\varphi_i, \quad B_i \mid i \in I\}$$

And vice versa.

We are effectively using the logical 'or' condition in each case to split up the pieces such that each piece only has one condition in the piecewise object representation. For example,

$$f(x) = \begin{cases} 5 & x \geq 5 \vee x \leq -5 \\ 0 & -5 < x < 5 \end{cases}$$

Is equivalent to:

$$f(x) = \begin{cases} 5 & x \geq 5 \\ 5 & x \leq -5 \\ 0 & -5 < x < 5 \end{cases}$$

Noting that the last piece cannot be split as it is a logical 'and'; that is, $x < 5 \wedge x > -5$.

3.7. Finally, there is no rule that says pieces are unique in a piecewise object (although generally redundant, duplicate pieces can be useful for grouping once more).

That is the following, for some set $J \subseteq I$

$$\phi = \{\varphi_i, \quad C_i \mid i \in I\}$$

is equivalent to:

$$\phi = \{\varphi_i, \quad C_i \mid i \in I\} \cup \{\varphi_j, \quad C_j \mid j \in J\}$$

The reason for this comes down to our well-definition argument as before (equal values).

3.6 Functions on piecewise objects

Finally! The fun begins.

3.8. Given the following piecewise object:

$$\phi = \{\varphi_i, \quad C_i \mid i \in I\}$$

We have that:

$$f(\phi) = \{f(\varphi_i), \quad C_i \mid i \in I\}$$

A quick intuition/proof is as follows: For all $i \in I$ suppose that at least one C_i is true. Then we have that $\phi = \varphi_i \implies f(\phi) = f(\varphi_i)$. By definition that means $f(\phi)$ is as above.

4 Common Piecewise Functions