

# A guide to understand and build a sequencer for a rhythm game using Stepmania's **SSC** definitions

Pedro G. Bascoy  
pepo\_gonba@hotmail.com

March 4, 2023

This guide aims to assist anyone in creating a sequencer for a rhythm game. A sequencer is a software that manages two crucial aspects of any rhythm game:

1. The timing of when the notes should be tapped concerning the beginning of the song.
2. The placement of the notes at any given time.

In this guide, we will delve into these two aspects when introducing gimmicks in the game. Gimmicks are alterations in these two aspects that provide additional (usually aesthetic) possibilities to step makers. Specifically, we will examine the StepMania approach to defining the settings required to set up a sequencer. Each definition will be analyzed in its own separate section, with examples provided, and mathematical formalism that can be easily implemented in the programming language of your choice at the end of each section.

This document's sections will focus on two or, at most, three mathematical spaces and the gimmicks that allow transformation from one space to another.

## 1 From song time to sequencer time

### 1.1 Introduction

The SSC file provides two methods of artificially pausing the note scrolling when the song is playing. Those gimmicks are referred as **#STOPS** and **#DELAYS**. Although there is a subtle difference between them, they are essentially identical w.r.t. the effect that produces.

One STOP definition might look as follows:

```
#STOPS: 4=5,6=2;
```

Let us convert the definition into a JSON-like structure:

```
{
  [
    beat: 4,
    stop: 5
  ],
  [
    beat: 6,
    stop: 2
  ]
}
```

This **#STOPS** definition is telling us a couple of things.

1. At beat 4, stop the scrolling for 5 seconds. Then resume.
2. At beat 6, stop the scrolling for 2 seconds. Then resume.

Similarly, one DELAY definition could look like this:

```
#DELAYS: 4=5,6=2;
```

And after converting it into the friendly structure:

```
{
  [
    beat: 4,
    delay: 5
  ],
  [
    beat: 6,
    delay: 2
  ]
}
```

The `#DELAYS` definition above is telling us essentially the same story as the `STOPS` definition shown before, i.e.:

1. At beat 4, stop the scrolling for 5 seconds. Then resume.
2. At beat 6, stop the scrolling for 2 seconds. Then resume.

The difference between these two is that notes that lie exactly at the stop/delay beat, will need to be tapped before and after the waiting time for the stops and delays, respectively.

## 1.2 Challenge

We would like to have a pair of functions  $t_{(s)}, t_{(d)} : \mathbb{R} \rightarrow \mathbb{R}$  that would retrieve the song time after stops and delays (sequencer time) from the song time. Additionally, we would like to have two inverse functions of  $t_{(s)}, t_{(d)}$ ,  $t_{(s)}^{-1}$  for `STOPS` and  $t_{(d)}^{-1}$  for `DELAYS` so we are able to map from the sequencer time into the song time.

## 1.3 Solution

To do so, let us imagine that we have a function  $f : \mathbb{R} \rightarrow \mathbb{R}$  that given a beat, it calculates the song time. To simplify things further, imagine that this song is has constant BPM of 60, so each second is worth 1 beat. Then  $f(b) = b$ , being  $b$  a beat.

Having this in mind, then we can write the piecewise functions

$$t_{(s)}(x) = t_{(d)}(x) = \begin{cases} x, & \text{if } x \leq 4; \\ 4, & \text{if } 4 < x \leq 4 + 5; \\ x - 5, & \text{if } 4 + 5 < x \leq 6 + 5; \\ 6 + 5, & \text{if } 6 + 5 < x \leq 6 + 5 + 2; \\ x - 2 - 5, & \text{if } x > 6 + 5 + 2; \end{cases} \quad (1)$$

that will map from song time to sequencer time for the `STOPS` and `DELAYS`, respectively. A plot of this function can be see in Figure 1.

We can also easily calculate the two different inverse functions to model when notes at stops or delays should be tapped w.r.t. the song time. On the one hand side, we write the function

$$t_{(s)}^{-1}(x) = \begin{cases} x, & \text{if } x \leq 4; \\ x + 5, & \text{if } 4 < x \leq 6; \\ x + 5 + 2, & \text{if } x > 6; \end{cases} \quad (2)$$

to map from the sequencer time into song time for `STOPS`. Similarly, the function

$$t_{(d)}^{-1}(x) = \begin{cases} x, & \text{if } x < 4; \\ x + 5, & \text{if } 4 \leq x < 6; \\ x + 5 + 2, & \text{if } x \geq 6; \end{cases} \quad (3)$$

will map from the sequencer time into the song time for `DELAYS`. Note that the signs at the conditions are slightly different.

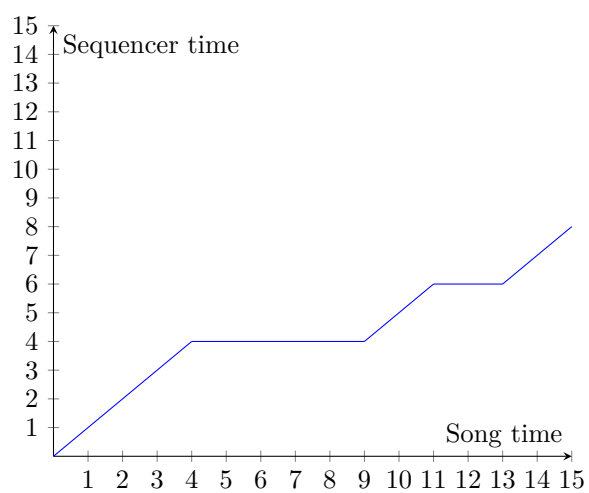


Figure 1: Plot of  $t_{(d)}$

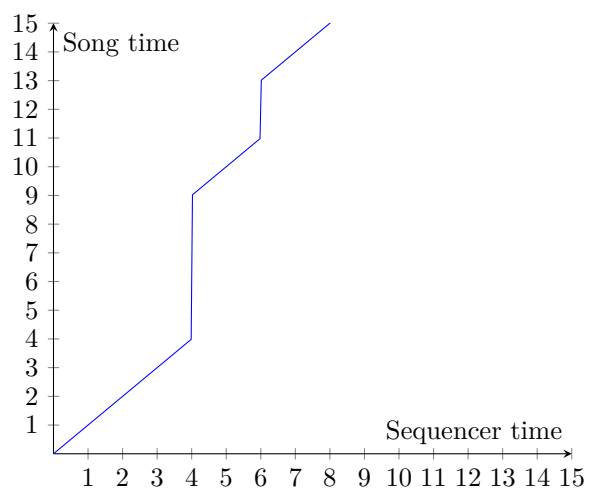


Figure 2: Plot of  $t^{-1}$

## 1.4 Formalization

Let  $\mathcal{T} = \left\{ \left( b_i^{(t)}, r_i \right) \right\}_{i=1}^n$  be a sequence of STOPS or DELAYS, where  $r_i$  is the stop or delay (measured in seconds) at beat  $b_i^{(t)}$ . Let  $f^{-1} : \mathbb{B} \rightarrow \mathbb{S}^*$  be a function that retrieves the second from the start of the song given a current beat, where  $\mathbb{B} = \mathbb{R}$ , and  $\mathbb{S}^* = \mathbb{R}$ .

We define a new set

$$\mathcal{T}' = \{(c_i, r_i)\}_{i=1}^n = \left\{ \left( f^{-1} \left( b_i^{(t)} \right), r_i \right) \right\}_{i=1}^n \quad (4)$$

where  $c_i$  is the second from the start of the song of beat  $b_i^{(t)}$ . We define the functions  $t_{(s)} : \mathbb{D} \rightarrow \mathbb{S}^*$  and  $t_{(d)} : \mathbb{S} \rightarrow \mathbb{D}$

$$t_{(s)}(x) = t_{(d)}(x) = \begin{cases} x - \sum_{j=1}^{i-1} r_j, & \text{if } c_{i-1} + \sum_{j=1}^{i-1} r_j < x \leq c_i + \sum_{j=1}^{i-1} r_j, \quad \forall i = 1, \dots, n; \\ c_i, & \text{if } c_i + \sum_{j=1}^{i-1} r_j < x \leq c_i + \sum_{j=1}^i r_j, \quad \forall i = 1, \dots, n; \\ x - \sum_{j=1}^n r_j & \text{if } x > c_n + \sum_{j=1}^n r_j; \end{cases} \quad (5)$$

when  $\mathcal{T}' \neq \emptyset$ , where  $c_0 := -\infty$ , that maps from the song time into the sequencer time.

We define the function  $t_{(s)}^{-1} : \mathbb{S}^* \rightarrow \mathbb{D}$

$$t_{(s)}^{-1}(x) = \begin{cases} x, & \text{if } x \leq c_1; \\ x + \sum_{j=1}^i r_j, & \text{if } c_i < x \leq c_{i+1}, \quad \forall i = 1, \dots, n; \end{cases} \quad (6)$$

with  $c_{n+1} := \infty$  which maps from the sequencer time into the song time for the STOPS, if  $\mathcal{T}' \neq \emptyset$ , and the function  $t_{(d)}^{-1} : \mathbb{D} \rightarrow \mathbb{S}$

$$t_{(d)}^{-1}(x) = \begin{cases} x, & \text{if } x < c_1; \\ x + \sum_{j=1}^i r_j, & \text{if } c_i \leq x < c_{i+1}, \quad \forall i = 1, \dots, n; \end{cases} \quad (7)$$

that maps from the sequencer time into the song time for the DELAYS, if  $\mathcal{T}' \neq \emptyset$ . If  $\mathcal{T}' = \emptyset$ , then  $t_{(s)}(x) = t_{(s)}^{-1}(x) = x$  and  $t_{(d)}(x) = t_{(d)}^{-1}(x) = x$ , for STOPS and DELAYS respectively. The final mapping from song time into sequencer time is the composition of  $t_{(d)}$  and  $t_{(s)}$ ,  $t_{(d)} \circ t_{(s)}$ . Similarly, the mapping from sequencer time into the song time is  $t_{(s)}^{-1} \circ t_{(d)}^{-1}$ .

## 2 From sequencer time to beat

### 2.1 Introduction

A SSC file gives a list of pairs which defines the bpm's. The first item in the pair is the target beat, and the second item is the desired BPM from that beat on. Let us imagine we have a SSC file with the following definition:

```
#BPMS:0=120,8=70,13=200;
```

Let us convert this cumbersome definition into a friendly structure:

```
{
  [
    beat: 0,
    bpm: 120
  ],
}
```

```

[
  beat: 8,
  bpm: 180
],
[
  beat: 13,
  bpm: 60
]
}

```

This #BPMS definition is telling us three things:

1. From beat  $-\infty$  to beat 8, the BPM is 120.
2. From beat 8 to beat 13, the BPM is 180.
3. From beat 13 to beat  $+\infty$ , the BPM is 60.

## 2.2 Challenge

We want to find a function  $f : \mathbb{R} \rightarrow \mathbb{R}$  that retrieves the current beat given the a second in the sequencer time space. This function is useful when a song is playing and we want to know at what beat we are at if we know how much time has passed since the start of the song. Also, as we will see later on, notes scroll at a BPM rate, so if we can have the inverse function of  $f$ ,  $f^{-1}$ , we can sort of know when the steps should be tapped as well.

## 2.3 Solution

First, let us convert BPMS to BPSS (Beats Per Second), since we are going to provide the input in seconds instead of minutes. We can do so by dividing the BPMS by 60, i.e.

$$\text{BPS}(x) = x \times \frac{\text{Beats}}{\text{Minute}} = x \times \frac{1 \times \text{Minute}}{60 \times \text{Seconds}} \frac{\text{Beats}}{\text{Minute}} = \frac{x}{60} \times \frac{\text{Beats}}{\text{Second}}. \quad (8)$$

Next, let us define a piecewise function  $f' : \mathbb{R} \rightarrow \mathbb{R}$  that gives the current BPS given the current Beat. Taking the #BPMS toy example from the previous section, we get that

$$f'(x) = \begin{cases} 2, & \text{if } x \leq 8; \\ 3, & \text{if } 8 < x \leq 13; \\ 1, & \text{if } x > 13. \end{cases} \quad (9)$$

In Figure 3 you can see the plot of  $f'$  we just defined in (9).

Note that by using  $f'$ , we can get the BPS at any beat of the song. This is great, but it does not quite solve our problem.

Next, we can calculate the SPB (Seconds Per Beat) by just inverting the BPS, i.e.

$$\text{SPB} = \frac{1}{\text{BPS}}, \quad (10)$$

and therefore we can define a function  $t : \mathbb{R} \rightarrow \mathbb{R}$

$$t(x) = x \times \text{SPB} \quad (11)$$

that given a beat  $x$  retrieves the current second.

Let

$$f^{-1}(x) = \begin{cases} \frac{x}{2}, & \text{if } x \leq 8; \\ \frac{8}{2} + \frac{x-8}{3}, & \text{if } 8 < x \leq 13; \\ \frac{8}{2} + \frac{5}{3} + x - 13, & \text{if } x > 13; \end{cases} \quad (12)$$

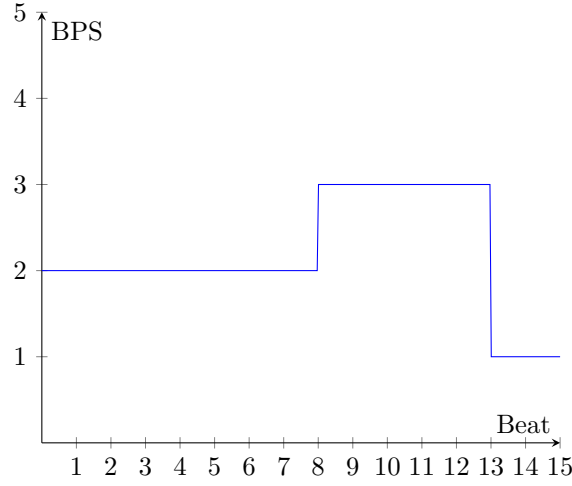


Figure 3: Plot of  $f'$

be the function that given a beat  $x$  retrieves the current second. This function is the result of plugging (10) and (11) into (9), and can be rewritten recursively as

$$f^{-1}(x) = \begin{cases} \frac{x}{2}, & \text{if } x \leq 8; \\ f^{-1}(8) + \frac{x-8}{3}, & \text{if } 8 < x \leq 13; \\ f^{-1}(13) + x - 13, & \text{if } x > 13. \end{cases} \quad (13)$$

Figure 4 depicts the function  $f^{-1}$ .

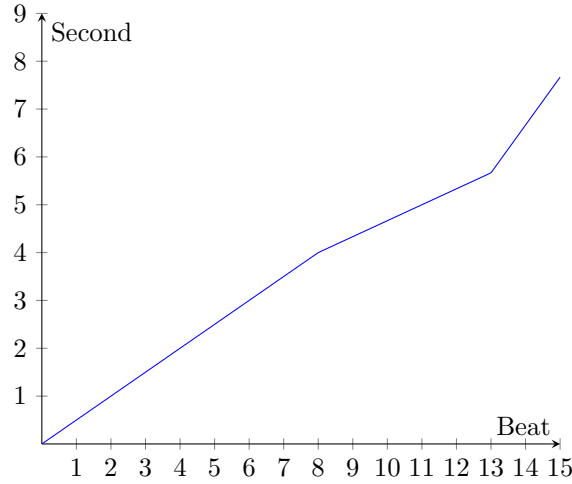


Figure 4: Plot of  $f^{-1}$

As it turns out, the function  $f$  that we are looking for is just the inverse function of  $f^{-1}$ , thus

$$f(x) = \begin{cases} 2x, & \text{if } x \leq 4; \\ (x-4) \times 3 + 8, & \text{if } 4 < x \leq 5.6; \\ x - 5.6 + 13, & \text{if } x > 5.6. \end{cases} \quad (14)$$

A plot of  $f$  can be seen in Figure 5.

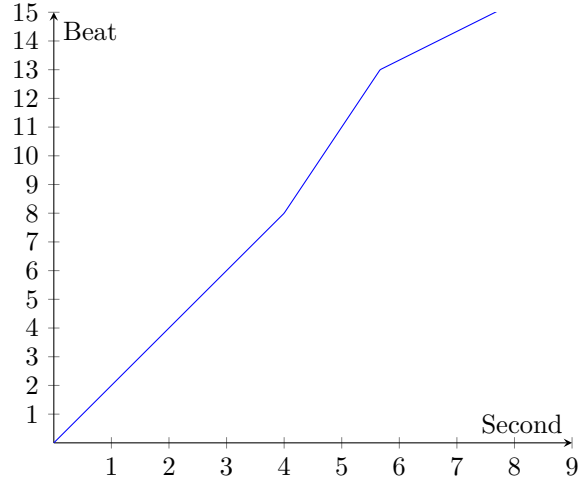


Figure 5: Plot of  $f$

## 2.4 Formalization

Let  $\{(b_i, v_i)\}_{i=1}^n$  be a sequence of  $n$  beat signatures, where  $v_i$  is the BPS value at beat  $b_i$ . Let  $f^{-1} : \mathbb{B} \rightarrow \mathbb{S}^*$  be a function that provided a beat, returns the seconds in the sequencer time space. We define this function as a  $n$ -step piecewise function

$$f^{-1}(x) = \begin{cases} \frac{x}{v_1}, & \text{if } x \leq b_2 ; \\ f^{-1}(b_i) + \frac{x - b_i}{v_i}, & \text{if } b_i < x \leq b_{i+1}; \quad \forall i = 2, \dots, n; \end{cases} \quad (15)$$

where  $b_1 := 0$ , and  $b_{n+1} := \infty$ .

Analogously, let  $f : \mathbb{S}^* \rightarrow \mathbb{B}$  be a function that provided a second in the sequencer time space, returns the beat from the zero second. We define this function as a  $n$ -step piecewise function

$$f(x) = \begin{cases} v_1 x, & \text{if } x \leq f^{-1}(b_2) ; \\ [x - f^{-1}(b_i)] \times v_i + b_i, & \text{if } f^{-1}(b_i) < x \leq f^{-1}(b_{i+1}); \quad \forall i = 2, \dots, n. \end{cases} \quad (16)$$

## 3 From beat to note position

### 3.1 Introduction

There are a pair of stepmania definitions that influence the position where notes should be placed on the screen. One of them is **#BPMS**, which arguably is the rate at what notes travel upwards towards the receptor w.r.t. to the music rhythm. We have already dealt with it in the previous section.

However, there is another gimmick that plays a role in the note positioning: **#SCROLLS**. Let's have a look at an example:

```
#SCROLLS:0=1,4=0,10=2;
```

Again, let us convert this cumbersome definition into a friendly structure:

```
{
  [
    beat: 0,
    scroll: 1
  ],
  [
    beat: 4,
    scroll: 0
  ],
}
```

```

[
  beat: 10,
  scroll: 2
]
}

```

The scroll gimmick changes the effective BPM at a given beat by a rate defined by the scroll value. Thus, in this example, the SCROLLS gimmick is changing the BPMs as follows:

1. From beat 0 to beat 4, the BPM is its value times 1.
2. From beat 4 to beat 10, the BPM is its value times 0. (all the steps in between this beats, will have the same position)
3. From beat 10 on, the BPM is its value times 2.

### 3.2 Challenge

We would like to have a function  $p : \mathbb{R} \rightarrow \mathbb{R}$  that given a beat, it retrieves the position w.r.t. the origin (or where the receptor is) where a note at that beat should be drawn.

### 3.3 Solution

Let us define a function  $p : \mathbb{R} \rightarrow \mathbb{R}$  that given a beat, retrieves the effective beat (i.e., beat with applied scrolls). For that matter, we just need to check out in what beats the scroll is taking place, and change the beat accordingly to the scroll rate. For our toy example the resultant  $p$  function looks like this

$$p(x) = \begin{cases} x, & \text{if } x \leq 4; \\ (x - 4) \times 0 + p(4), & \text{if } 4 < x \leq 10; \\ (x - 10) \times 2 + p(10), & \text{if } x > 10. \end{cases} \quad (17)$$

The function  $p$  is depicted in Figure 6. Now, this is great! By asking  $p$ , now we have the effective

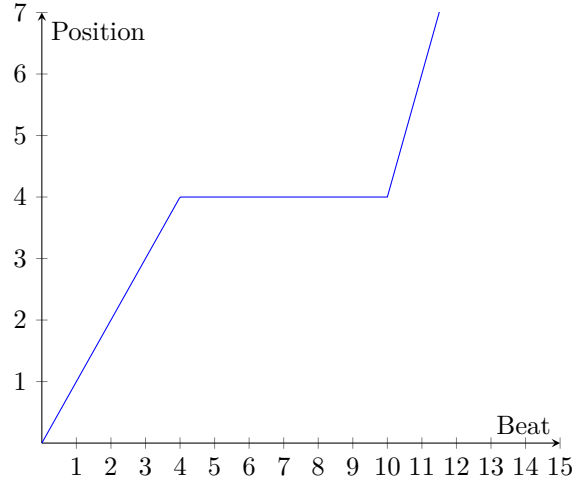


Figure 6: Plot of  $p$

beat, and therefore the position. Note that the drawing position IS NOT the beat for that we are going to use to calculate when the note is needed to be tapped!

### 3.4 Formalization

Let

$$\left\{ \left( b_i^{(s)}, s_i \right) \right\}_{i=1}^m = S = B^{(s)} \times S \quad (18)$$



be a sequence of  $m$  scroll signatures, where  $s_i \in S = \{s_j\}_{j=1}^m$  is the scroll value at beat  $b_i^{(s)} \in B^{(s)} = \{b_j^{(s)}\}_{j=1}^m$ .

We define the function  $p : \mathbb{R} \rightarrow \mathbb{R}$

$$p(x) = \int g(x)dx = \begin{cases} s_1 x, & \text{if } x \leq b_2^{(s)}; \\ p(b_i^{(s)}) + (x - b_i^{(s)}) \times s_i, & \text{if } b_i^{(s)} < x \leq b_{i+1}^{(s)}; \quad \forall i = 2, \dots, n', \end{cases} \quad (19)$$

as the function that retrieves the position given a beat.

## 4 From beat to scroll

### 4.1 Introduction

Another issue that we might have is to know how far upwards we should scroll the notes from its original position given the current beat. If we did not have any other gimmicks, this would be very easy to compute. Let us suppose that the notes rendered in the screen are squares of one unity of length and height, and that one beat is worth one distance of separation, as shown in Figure 7. In this set up,

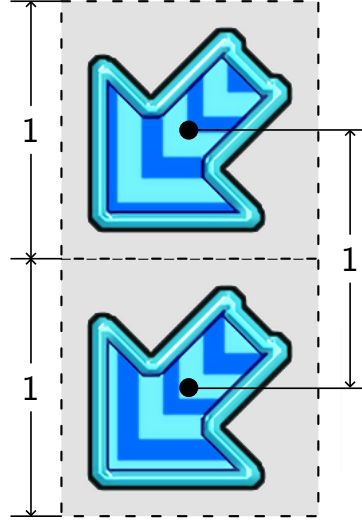


Figure 7: Two notes inside with their respective bounding boxes separated apart by one beat.

the amount of scroll that we have to apply for a given beat  $x$  is just  $-x$  (if we were scrolling upwards, on the  $y$  axis).

However there are three gimmicks that modify this scroll function in a number of different ways: #WARPS, #SPEEDS, and #SCROLLS. Let us see an example to know what these modifiers do. Suppose we have the following #WARPS definition:

```
#WARPS:4=2;
```

It is equivalent to this one:

```
{
  [
    beat: 4,
    warp: 2
  ]
}
```

As it turns out, this gimmick is telling us the following information:

1. At beat 4, warp over the next 2 following beats, i.e. skip two beats.

2. Notes with beats in the range  $[4, 4 + 2) = [4, 6)$  will become fake notes.

On the other hand, suppose we have the following **#SPEEDS** definition:

```
#SPEEDS:4=2=1=0,6=0.5=1=1;
```

which is equivalent to this one:

```
{
  [
    beat: 0,
    speed: 1,
    span: 0,
    type: 0
  ],
  [
    beat: 6,
    speed: 7,
    span: 1,
    type: 1
  ]
}
```

Well, the information provided with this gimmick is the following:

1. From beat 0 on, the separation between notes increases by a factor of 1, the scrolling speed also increases by a factor of 1, and because the **type** is 0, this transition is smoothly applied in the span of 0 **beats**.
2. From beat 6 on, the separation between notes increases by a factor of 7, the scrolling speed also increases by a factor of 7, and because the **type** is 1, this transition is smoothly applied in the span of 1 **second**.

We have already talked about how the **#SCROLLS** definition changes where the notes should be placed on the screen. It turns out that also has an impact on the scrolling function. Taking the same example from Section 3, we can extract the following information:

1. From beat 0 to beat 4, the scrolling 1 times as fast.
2. From beat 4 to beat 10, the scrolling is 0 times as fast.
3. From beat 10 on, the the scrolling is 2 times as fast.

## 4.2 Challenge

We would like to have a function  $q : \mathbb{R} \rightarrow \mathbb{R}$  that calculates the warped beat value for a given beat as well as its inverse,  $q^{-1}$ , a function  $e : \mathbb{R} \rightarrow \mathbb{R}$  that calculates the speed factor at a given beat, and a function  $g : \mathbb{R} \rightarrow \mathbb{R}$  that calculates the scroll value.

## 4.3 Solution

**Warps** First, let us deal with the problem of the WARPS. We would need to create a function that maps from beats to warped beats first, so we could retrieve the warped beat value w.r.t. it. Actually, the function that we are looking for is pretty much identical to that shown in Figure 2

$$q(x) = \begin{cases} x, & \text{if } x \leq 4; \\ x + 2, & \text{if } x > 4. \end{cases} \quad (20)$$

You can see the function plotted in Figure 8.

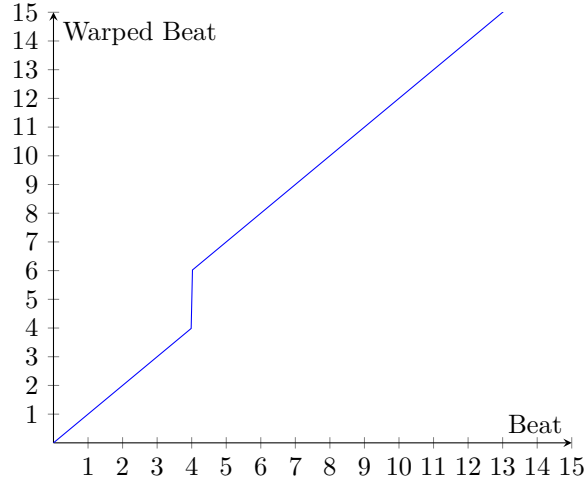


Figure 8: Plot of  $q$

And that is that. Actually, the output of this function is already the scroll value that we are looking for! It is inverse

$$q^{-1}(x) = \begin{cases} x, & \text{if } x \leq 4; \\ 4, & \text{if } 4 < x \leq 4 + 2; \\ x - 2, & \text{if } x > 4 + 2; \end{cases} \quad (21)$$

also is very similar to how we calculated  $t_{(s)}^{-1}$  in Section 1. You can visualize it in Figure 9.

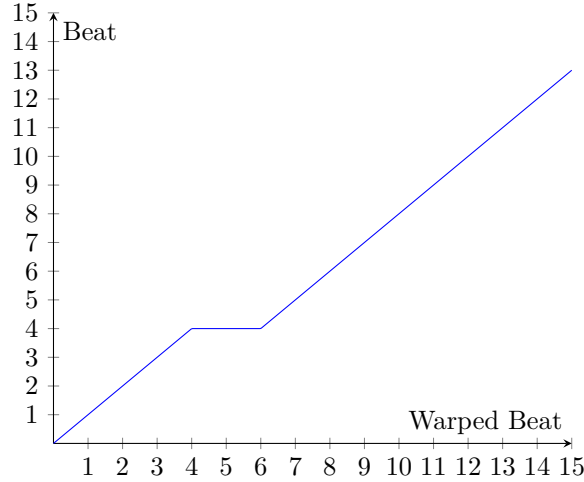


Figure 9: Plot of  $q^{-1}$

**Speeds** The function that we need to come up for the speeds is simple as well. However, it gets a bit complicated when dealing with speeds of **type 1**: Since we are working at the beat space, we need somehow to convert a span of seconds into a span of beats so we can perform operations in the same units. We do so by mapping a given beat into the song time space, then add the span of seconds, and then mapping back into the beat space. To keep things simple, let us imagine that we are running a 60 BPM song with no delays or stops. In this particular case, we can write

$$e(x) = \begin{cases} 1, & \text{if } x \leq 6; \\ \frac{7-1}{1}(x-6) + 1, & \text{if } 6 < x \leq 6 + 1; \\ 7, & \text{if } x > 6 + 1. \end{cases} \quad (22)$$

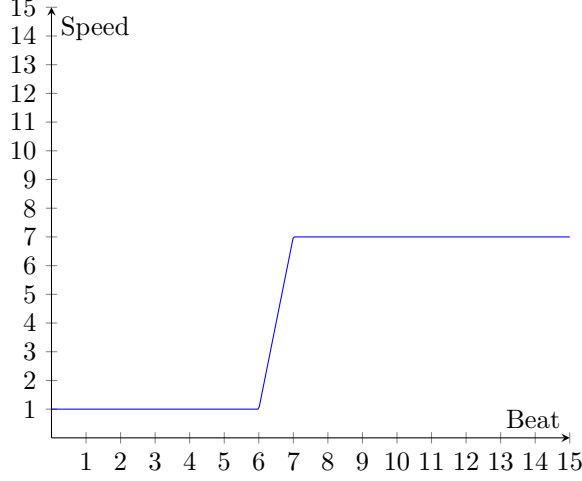


Figure 10: Plot of  $e$

**Scrolls** There is not much to say here. The function  $g$  that we are looking for is just identical to the function  $p$ ,

$$g(x) = p(x) \quad (23)$$

#### 4.4 Formalization

**Warps** Let  $\mathcal{W} = \left\{ \left( b_i^{(w)}, w_i \right) \right\}_{i=1}^n$  be a sequence of WARPS, where  $w_i$  is the warp (measured in beats) at beat  $b_i^{(w)}$ .

We define the function  $q : \mathbb{B} \rightarrow \mathbb{T}$

$$q(x) = \begin{cases} x, & \text{if } x \leq b_1^{(w)}; \\ x + \sum_{j=1}^i w_j, & \text{if } b_i^{(w)} < x \leq b_{i+1}^{(w)}, \quad \forall i = 1, \dots, n; \end{cases} \quad (24)$$

with  $b_{n+1}^{(w)} := \infty$  which maps from the beat space into the translation space, if  $\mathcal{W} \neq \emptyset$ .

We define the function  $q^{-1} : \mathbb{T} \rightarrow \mathbb{B}$

$$q^{-1}(x) = \begin{cases} x - \sum_{j=1}^{i-1} w_j, & \text{if } b_{i-1}^{(w)} + \sum_{j=1}^{i-1} w_j < x \leq b_i^{(w)} + \sum_{j=1}^{i-1} w_j, \quad \forall i = 1, \dots, n; \\ b_i^{(w)}, & \text{if } b_i^{(w)} + \sum_{j=1}^{i-1} w_j < x \leq b_i^{(w)} + \sum_{j=1}^i w_j, \quad \forall i = 1, \dots, n; \\ x - \sum_{j=i}^n w_j & \text{if } x > b_n^{(w)} + \sum_{j=1}^n w_j; \end{cases} \quad (25)$$

when  $\mathcal{W} \neq \emptyset$ , where  $b_0^{(w)} := -\infty$ , that maps from the translation space into the beat space.

If  $\mathcal{W} = \emptyset$ , then  $q(x) = q^{-1}(x) = x$ .

**Speeds** Let  $\mathcal{E} = \left\{ \left( b_i^{(e)}, s_i, p_i, t_i \right) \right\}_{i=1}^n$  be a sequence of SPEEDS, where  $s_i$  is the speed change at beat  $b_i^{(e)}$  with a span transition of  $p_i$  beats if  $t_i = 0$ , or  $p_i$  seconds if  $t_i = 1$ .

We define a new sequence  $\mathcal{E}' = \left\{ \left( b_i^{(e)}, s_i, p'_i \right) \right\}_{i=1}^n$  from  $\mathcal{E}$  as

$$\mathcal{E}' = \bigcup_{i=1}^n \begin{cases} \left\{ \left( b_i^{(e)}, s_i, p_i \right) \right\}, & \text{if } t_i = 0; \\ \left\{ \left( b_i^{(e)}, s_i, g(p_i, b_i^{(e)}) \right) \right\}, & \text{otherwise,} \end{cases} \quad (26)$$

where

$$g(p, b) = (f \circ t_{(s)} \circ t_{(d)}) \left( (t_{(d)}^{-1} \circ t_{(s)}^{-1} \circ f^{-1}) (b) + p \right). \quad (27)$$

We define the function  $e : \mathbb{B} \rightarrow \mathbb{E}$

$$e(x) = \begin{cases} s_1, & \text{if } x \leq b_1^{(e)}; \\ \frac{s_i - s_{i-1}}{p'_i} (x - b_i^{(e)}) + s_{i-1}, & \text{if } b_i^{(e)} < x \leq b_i^{(e)} + p'_i \wedge p'_i \neq 0, \quad \forall i = 1, \dots, n; \\ s_i, & \text{if } b_i^{(e)} + p'_i < x \leq b_{i+1}^{(e)}, \quad \forall i = 1, \dots, n, \end{cases} \quad (28)$$

where  $b_{n+1}^{(e)} := \infty$ , and  $s_0 := s_1$ .

**Scrolls** We define  $g : \mathbb{R} \rightarrow \mathbb{R}$

$$g(x) = p(x) \quad (29)$$

where  $p$  is (19).

## 5 Positioning and scrolling notes

Alright! Now we have formally defined everything we need to build a sequencer using stepmania's notation. This section will show how to use all functions defined formally in the sections above to determine where to draw the notes in the screen at any given song time.

To ease the understanding, Table 1 gathers all the functions that we are going to need as well as a brief description for each one of them.

Symbol	Description
$\mathbb{S}$	Song time space (s) $\mathbb{S} = \mathbb{R}$ .
$\mathbb{D}$	Delayed time space (s) $\mathbb{D} = \mathbb{R}$ .
$\mathbb{S}^*$	Sequencer time space (s) $\mathbb{S}^* = \mathbb{R}$ .
$\mathbb{B}$	Beat space (beats) $\mathbb{B} = \mathbb{R}$ .
$\mathbb{P}$	Position space (units) $\mathbb{P} = \mathbb{R}$ .
$\mathbb{T}$	Scroll space (units) $\mathbb{T} = \mathbb{R}$ .
$\mathbb{E}$	Speed space (speed factor units) $\mathbb{E} = \mathbb{R}$ .
$\mathbb{W}$	Warped Beat space (units) $\mathbb{W} = \mathbb{R}$ .
$t_{(d)}$	Function $t_{(s)} : \mathbb{S} \rightarrow \mathbb{D}$ that maps from the song time space into the delayed time space.
$t_{(d)}^{-1}$	Function $t_{(d)}^{-1} : \mathbb{D} \rightarrow \mathbb{S}$ that maps from the the delayed time space into the song time space.
$t_{(s)}$	Function $t_{(s)} : \mathbb{D} \rightarrow \mathbb{S}^*$ that maps from the delayed time space into the sequencer time space.
$t_{(s)}^{-1}$	Function $t_{(s)}^{-1} : \mathbb{S}^* \rightarrow \mathbb{D}$ that maps from the the sequencer time space into the delayed time space.
$f$	Function $f : \mathbb{S}^* \rightarrow \mathbb{B}$ that maps from the sequencer time space into the beat space.
$f^{-1}$	Function $f^{-1} : \mathbb{B} \rightarrow \mathbb{S}^*$ that maps from the beat space into the sequencer time space.
$p$	Function $p : \mathbb{B} \rightarrow \mathbb{P}$ that maps from the beat space into the position space.
$q$	Function $q : \mathbb{B} \rightarrow \mathbb{W}$ that maps from the beat space into the warped beat space.
$q^{-1}$	Function $q^{-1} : \mathbb{W} \rightarrow \mathbb{B}$ that maps from the warped beat space into the beat space.
$e$	Function $e : \mathbb{B} \rightarrow \mathbb{E}$ that maps from the beat space into the speed space.
$g$	Function $g : \mathbb{W} \rightarrow \mathbb{T}$ that maps from the warped beat space into the scroll space.

Table 1: Table of functions and spaces.

We will take the following assumptions:

1. We are modeling a rhythm game whose notes scroll upwards.
2. The receptor is placed at the origin of the scrolling axis.
3. The beat 0 has a position 0 in the scrolling axis.

Let  $\mathcal{N} = \{(u_i)\}_{i=1}^n$  be a sequence of notes where  $u_i$  is the beat when the  $i$ -th note should be tapped. We define a new set

$$\mathcal{N}' = \{(u_i, v_i, w_i)\} = \bigcup_{\{u\} \in \mathcal{N}} \{(u, g(u), p(u))\} \quad (30)$$

where  $w : \mathbb{W} \rightarrow \mathbb{S}$

$$w(x) = \left( t_{(d)}^{-1} \circ t_{(s)}^{-1} \circ f^{-1} \circ q^{-1} \right) (x), \quad (31)$$

$v_i$  is the exact time where the  $i$ -th note should be tapped, and  $w_i$  is its relative position at beat 0 in the scrolling axis. At a given moment in time  $t$  w.r.t. to the begginig of the song, the  $i$ -th note should be positioned at

$$\left[ w_i - \left( g \circ q \circ f \circ t_{(s)} \circ t_{(d)} \right) (t) \right] \times \left( e \circ q \circ f \circ t_{(s)} \circ t_{(d)} \right) (t). \quad (32)$$