Non-linear Sequencing

Michele Zaccagnini
michelezaccagnini@hotmail.com

Introduction

Since the early developments of the MIDI protocol, electronic musicians have been making use of sequencers and keyboards to playback MIDI data. In its most basic form, a sequencer triggers events by reading through a list of performance information, such as pitch, duration, and velocity. While a keyboard allows for direct human-machine interaction, a sequencer presupposes a mediated interaction in which the sequence is first written and stored and subsequently played back by the sequencer. The sequencer has become a ubiquitous tool for music composition and can be found in virtually any Digital Audio Workstation (DAW) (Cuadrado, 2015).[1] Furthermore, the sequencer has become a means "for musical expression, a vehicle for communicating music ideas, and a clear and articulate musical voice." (Brown, 2007: 10) as well as expanded the possibilities for modern composers (Théberge 2012: 82). In this paper, I discuss how the core time function at the base of the sequencer can be manipulated and turned into a compositional parameter. In particular, the paper explores ways of breaking away from the linear nature of the sequencer's time function, one that has characterized it from its inception. The procedures and tools presented in this paper are part of an evolving compositional practice that goes under the name of Nonlinear Sequencing (NLS); the tools that implement NLS are currently developed in Max 8. The procedures present interesting implications for electronic musicians and audio programmers interested in generative music, video game music and interactivity (Plut, 2020).

Nonlinear Sequencing: Goals

Traditionally musical sequencers employ linear time-ramps to trigger a sequence of musical events. The sequence can be thought of as an ordered list of events, each event being time-stamped so that the sequencer triggers its output whenever a certain point in time is reached. The time function driving the sequencer can be assumed as a sawtooth oscillator of variable frequency and with range [0, 1]. A sequencer's time

[1] 'A concentration process has come about around the computer, substituting physical elements (analogue or digital hardware) for virtual elements (software programs and plugins), building a new creation and production ecosystem usually centralized in the sequencer, workstation or DAW.'

function can be formalized as the fractional component of the running time t multiplied by its frequency value.

$$S(x) = mod(xt, 1) \quad [1]$$

Where of x is the sequencer's frequency.

NLS has 3 fundamental goals:

1. Create a systematic approach to explore other, nonlinear, possibilities for $S(x)$
2. Maintain overall predictability of the sequencer behavior over time: the points in time where the sequencer wraps up must be predictable with maximum precision
3. Creating a computationally efficient and sample accurate DSP pipeline

Point (1) deals with the question of how to apply nonlinearity to the sequencer's process. Point (2) deals with the issue of maintaining predictability over time of the sequencer's behavior once the linearity of the function is affected. Based on the hypothesis formulated in (1) and (2), point (3) tackles the implementation NLS in an audio programming environment such as Max 8.

Exploring ways of creating NLS

In music programming, the sequencer's frequency can be either manipulated

1. Continuously, e.g. by an LFO,
2. Discretely, e.g. by direct intervention of the user.

In (1) the sequencer's behavior can be either linear (e.g. triangle LFO) or nonlinear (e.g. sinusoidal LFO).

In (2) the behavior of $S(x)$ is linear with some disruptions whenever a change happens.

Both approaches present limitations for the user.

- In (1) we have limited choices in terms of oscillator functions.
- In (2) the rate discontinuities make the behavior of the sequencer over time unpredictable.[2]

With the introduction of Nonlinear Sequencing, we take these two approaches as a starting point and propose ways of solving synchronization problems in (2) and adding more options in terms of functions in (1), including customized functions.

First case of NLS: Sequencer Distortion (SD)

The first instance of NLS is the easiest to grasp and implement as it simply expands on the idea of using nonlinear LFOs to control a sequencer. The Sequencer Distortion approach (SD) is borrowed from the sound synthesis process known as *waveshaping*.

[2] For instance, when using interlocking polyrhythms and multiple sequencers, changing one sequencer's rate independently from the other will entropically undermine the initial order.

Waveshaping is an 'approach to modulating a signal [allowing it] to pass through a suitably chosen nonlinear function.' (Puckette, 2007)
Practically speaking this approach simply adds a lookup table (LUT) to the sequencer's pipeline. The function contained in the lookup table is usually referred to as the *transfer function* (*ibid.*). While waveshaping proper is used to produce timbral distortions, SD creates distortions in the sequencer's ramp. A simple case of SD is a basic quadratic transfer function, with index [0, 1]

$$f(x) = x^2$$

So that [1] becomes

$$S(f(x)) = mod(x^2 t,\ 1) \qquad [2]$$

Producing and exponential 0-1 ramp.
Compared to its sound synthesis cousin, SD presents some idiosyncrasies.
While virtually any function can be stored in the LUT, the output of $S(f(x))$ could create uncharacteristic and undesirable behaviors in the sequencer. In particular, if the LUT's range is equal to the sequencer's range the pattern will be reproduced in its entirety[3] (e.g. [2]), but if the LUT's range is less than the sequencer's range, only part of the pattern will be output. Finally, if the LUT's range is greater than the sequencer's range there will be time gaps in the sequence's playback.

Second Case of NLS: Fluctuating Sequencer

The second case of NLS allows for the sequencer's frequency to change arbitrarily, while preserving a predictable behavior over time; I call this the Fluctuating Sequencer (FS). Strictly speaking, the sequencer's ramp stays linear throughout the FS process but its frequency is allowed to fluctuate from one ramp to the next producing an overall nonlinearity in the tempo treatment. Crucially, these fluctuations need to happen around a fixed frequency (f_c).[4]
Since one of the goals of NLS is to maintain a predictable behavior of the sequencer over time,[5] FS will have specific and predictable points in time at which it will align with a sequencer with frequency f_c.

[3] The resulting output can be rhythmically quite different from its linear correspondent. The timeline can potentially fold onto itself to have single events in the sequence's table appear more than once, a possibility that traditional sequencers are excluded from.
[4] Compared to SD, FS is more suitable for designing larger nonlinear rhythmic structures as it takes place over several cycles, instead of on a single sequencer's ramp.
[5] See p.2. Incidentally, the predictability attribute is crucial if one wants to use NLS for rhythmic layering

Therefore, we can build FS starting from its fixed-frequency alignment prerequisite. Given a constant frequency f_c and a set of fluctuating frequencies $[f_1, f_2, ..f_n]$, FS needs to satisfy the following equality:

$$nf_c = \sum_{1}^{n} f_n \qquad [3]$$

Which means that the total time that it takes a sequencer to run through n cycles at frequency f_c is equal to the time that it takes a sequencer to run for n cycles at frequencies $[f_1, f_2, ..f_n]$.

Since the goal of FS is to create oscillations around a fixed frequency, we should assume f_c as constant and find a way to change the set $[f_1, f_2, ..f_n]$ to satisfy [3]. One way to do that is to create a set of arbitrary frequencies of length $m = n/2$ and pair it with another list of equal length of "compensating" frequencies: for each f there has to be a f' allows for the fluctuating frequency to be compensated. The total length of the list is n .

$$f + f' = 2f_c \qquad [4]$$

From which

$$f' = 2f_c - f \qquad [5]$$

Since f' cannot be negative then the $f < 2f_c$ condition needs to be met.[6]

FS has, therefore, two inputs f_c and a list of [f_i] ; while the list of $[f'_i]$ is calculated by applying iteratively [5] to $[f_i]$. The output of FS is a set of ramps of variable frequency that wraps around at time f_c/n, at which point it will start repeating.

Implementation of NLS in Max

NLS is a heuristic that was borne out of a compositional necessity of creating rhythmic oscillations around a beat. Its real life implementation is robust and allows for signal-rate accuracy.[7]

Both SD and FS employ a top-to-bottom sample-rate pipeline, the only non-signal messages being the input information about frequency for SD and fixed-frequency and fluctuating frequencies for FS (assuming that the pattern is a fixed variable). They both share the same method for triggering sequence events based on a non-interpolating

[6] This condition makes sense intuitively: if the fluctuating time interval is more than twice the fixed time interval there is no small enough amount of time that will allow for *f'* to compensate for *f.*

[7] Earlier implementations of NLS employed some passages that relied on the Max scheduler; this lead to inaccuracies and inconsistent behaviors, undermining NLS' very purpose.

sample-playback object [index~] and a [buffer~] containing zeros except for trigger samples placed at specific points of value 1.

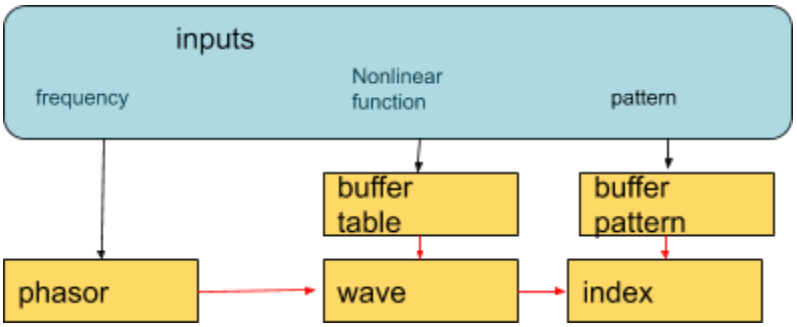

Fig. 1- Sequencer Distortion: Max diagram. Yellow boxes are signal rate objects.

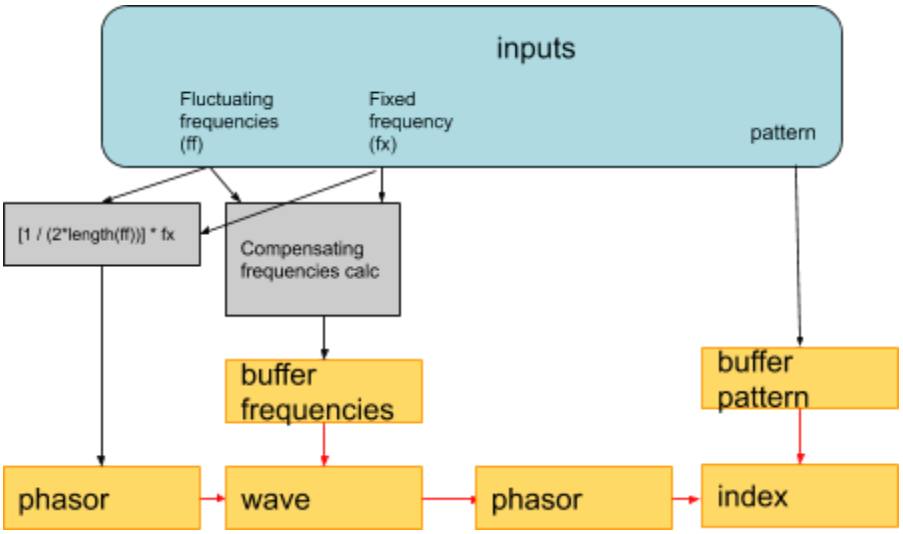

Fig. 2- Fluctuating Sequencer: Max diagram. Yellow boxes are signal rate objects.

Experimental use of Non-linear sequencing

The development of NLS is based on a heuristic approach to rhythmic perception as part of growing research on rhythms and meters (Brochard, 2003; Potter, 2008). One of

the aesthetic purposes of NLS is to explore the "grey areas" of meter perception[8]. To this end NLS can be helpful in its basic characteristic of shifting the focus of the process from the sequence to the sequencer's function. This way the same sequence can be manipulated in different ways while remaining unaltered in its underlying structure. NLS' focus on the bending of the timeline makes it an interesting candidate for the exploration of questions such as meter perception.

Conclusions

Electronic musicians and composers tend to focus their attention on the sequence, i.e. the composition, taking the linear behavior of the sequencer for granted. Certainly, many commercial DAWs implement sequencers that allow for tempo changes and occasionally they provide ways to bend the tempo via a nonlinear curve.[9] These changes operate locally, require a direct action by the composer and are generally approached as temporary disruptions of the sequencer's linearity, commonly perceived as *accelerandos* and *ritardandos*.

Nonlinear Sequencing (NLS) differs from the above mentioned approaches to tempo bending in that it turns the tempo function into one of the sequencer's parameters, affecting the entirety of the sequencer's behavior and not simply a suspension of the default linearity of the sequencer. While the underlying rigidity of a sequencer's process could seem a counterintuitive choice to achieve malleable meters, its simple and coherent structure is desirable when designing rhythmic algorithmic processes. NLS provides a way to get the best of both worlds: flexibility and robustness.

Example Patches

Example patches and tools can be shared upon request

References

Brochard, Renaud & Abecasis, Donna & Potter, Douglas & Ragot, Richard & Drake, Carolyn. (2003). The "Ticktock" of Our Internal Clock Direct Brain Evidence of

[8] Meter defined as : 'the tendency to periodically group sound events, perceiving an alternation of accented (''strong'') and unaccented (''weak'') beats, takes place even in perfectly regular sequences of identical tones.' (Potter, 2008)

[9] DAWs such as Logic, Reaper and Cubase allow for automations of the tempo of the session using linear and non linear ramps.

Subjective Accents in Isochronous Sequences. Psychological science. 14. 362-6. 10.1111/1467-9280.24441.

Brown, Andrew R. (2007), Computers in Music Education: Amplifying Musicality, New York: Routledge.

Cale Plut, Philippe Pasquier,Generative music in video games: State of the art, challenges, and prospects,Entertainment Computing, Volume 33,2020,100337,ISSN 1875-9521

Cuadrado, Francisco. (2015). The use of sequencer tools during the composition process: A field study. Journal of Music Technology and Education. 8. 55. 10.1386/jmte.8.1.55_1.

Potter, Douglas & Fenwick, Maggi & Abecasis, Donna & Brochard, Renaud. (2008). Perceiving rhythm where none exists: Event-Related Potential (ERP) correlates of subjective accenting. Cortex; a journal devoted to the study of the nervous system and behavior. 45. 103-9. 10.1016/j.cortex.2008.01.004.

Puckette, M. The Theory and Practice of Electronic Music
http://www.crca.ucsd.edu/~msp/techniques/v0.02/book-html/

Théberge, P. (1997), Any Sound You Can Imagine: Making Music/Consuming Technology, Middletown, CT: Wesleyan University Press.