



Hyperion is a semi-modular scaled algorithmic MIDI note generator suited for a wide range of applications.

Algorithmic generators can be used as performance tools, for the facile creation of sequences in composition, for accompaniment or inspiration.

The architecture is designed to quickly translate your ideas into parts. Familiarising yourself with the function of each element will assist in maximising your use of Hyperion.

#### contents:

introduction	2
clock	3
period	3
divider	4
lfos	5
sequencers	5
counters	6
step modifier	7
key	7
chord	7



output	8
count & reset	9
list of destinations	9
sequencing tips	10
license	11

	major (ionian)		leading whole tone
	dorian		lydian minor
	phrygian		overtone
	lydian		hindu
	mixolydian		locrian major
	aeolian (natural)		locrian modification
	locrian		algerian
	h. minor		hungarian gypsy
	m. minor		blues heptatonic 2
	whole tone		undoc heptatonic 2
	major pentatonic		enigmatic
	pentatonic variation		undoc heptatonic 3
	undoc penta 1		undoc heptatonic 4
	undoc penta 2		undoc heptatonic 5
	relative minor penta		double harmonic
	egyptian		phrygian minor
	shomyo		persian
	new pentatonic		undoc heptatonic 6
	undoc penta 3		neapolitan major
	pelog		javanese
	balinese		neapolitan minor
	japanese		xotic
	balinese 2		major bebop
	augmented hexatonic		major + dim. fifth
	undoc hexatonic 1		o. messiaen m. 2
	native american		o. messiaen M. 4
	mystic hexatonic		calliope
	undoc hexatonic 2		undoc octotonic 1
	undoc hexatonic 3		undoc octotonic 2
	blues heptatonic 1		o. messiaen m. 1
	hungarian major		spanish octotonic
	undoc heptatonic 1		chromatic

scales used in Hyperion

Hyperion's architecture is semi-modular, meaning that each component can be driven by a selection of other components precedent to it in the modulation hierarchy.

Each component has a source that it counts, and a source that resets it, for instance a sequencer might count measures and be reset by MIDI NoteOn ('gate') events.

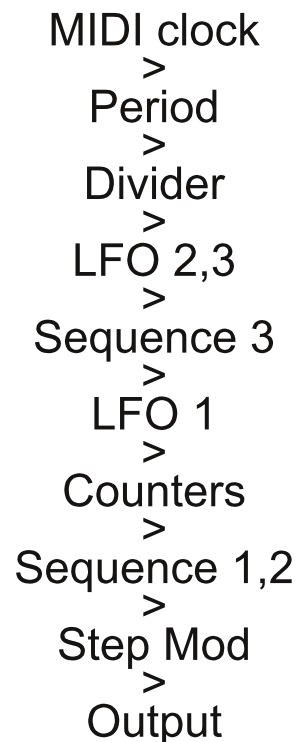
This allows you to structure your sequence in groups that can be modulated, ie. the 8th measure counts backwards et c.

Hyperion's output is recallable, and translates between scales and tempos. The output can be transposed using a MIDI keyboard and triggered with MIDI gate events, to function as a complex phrase arpeggiator.

The first section of the manual describes the components, the second section describes modulation routings to assist you in "thinking modular sequencing."

Have fun out there, and don't forget to hit record and send a snapshot back to earth every now and then :)

xoxos.

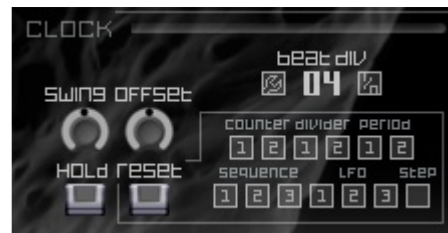


Modulation Hierarchy

## Clock

The basic sequence resolution is determined by the **beat div** setting, which indicates the number of clock pulses per beat. This parameter is accompanied by the **1/n** button, used to indicate slower rates such as 1/2 (notes per beat) and the **wrench** button, which holds the current setting when making an adjustment.

The clock section also features standard **swing** and **offset** knobs, which move the clock forward or backwards and both have a flat region in the center to assist in returning them to zero.



The **hold** button pauses the clock for as long as it is held.

The **reset** button reinitialises the selected items when it is held during a clock pulse.

## Period

The primary time divisions (eg. 'measure,' or 'phrase length') can be determined with the two period counters. These counters simply return one clock pulse every n pulses, for instance, a 'period' of 8 outputs a pulse every 8 clock pulses.

To make these basic counters more useful, they include an **offset** value. A simple illustration of this is open hats in house music, which are on every 2nd 8th note.. at a clock resolution of 16th notes, they could be described as a period of 4 with an offset of 2.

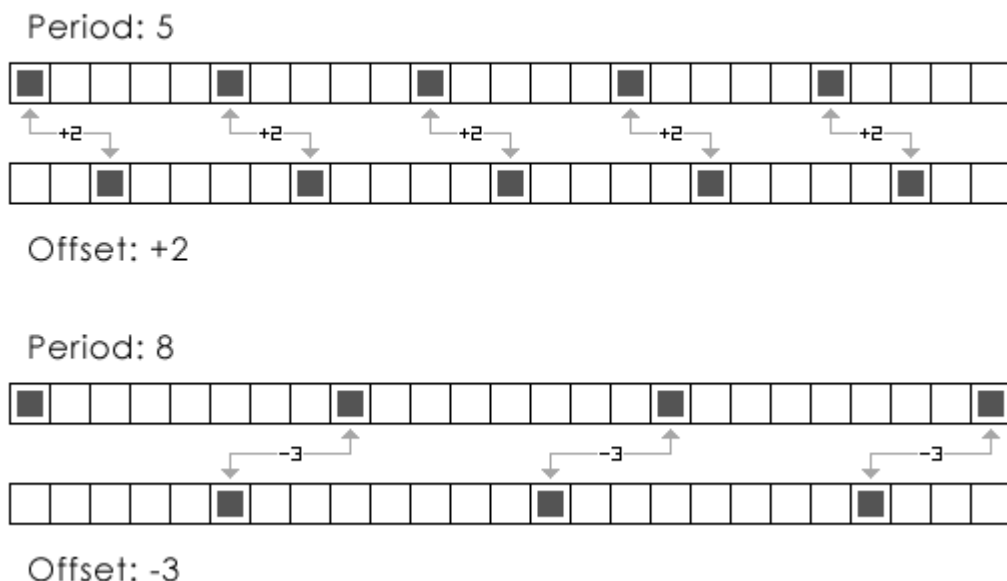
The period counters can be reset with the **beats** value, which simply resets them at the indicated number of host-synced beats. To turn sync off, set **beats** to 00.

Note that when the period counters are synced to a number of host beats, they pass the beats sync pulse along with their own output.

Activating the gate buttons syncs the period counters to the first clock pulse after a NoteOn event is received.



## Period Counter



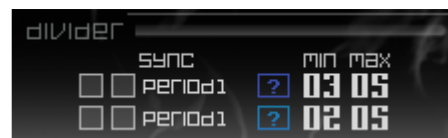
## Divider

The dividers are counters whose function is similar to the **period** counter, except that the interval of counting is randomised. The **min** and **max** values set the number of steps for each division.

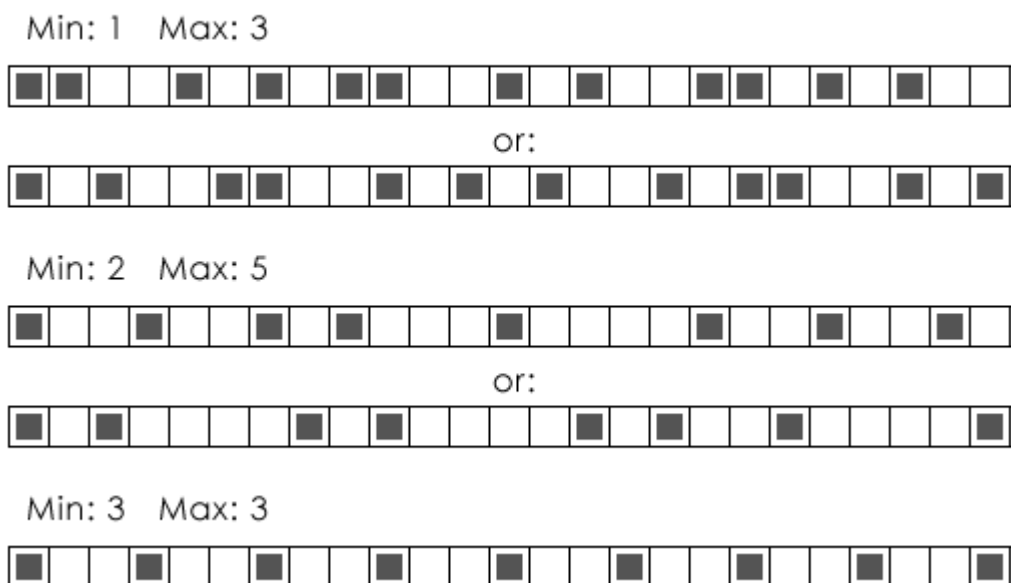
Dividers are used to create spacing or phrasing in a sequence.. they can be used to reset a sequence and make it repeat a short phrase, to add or remove gate events, or as the clock source for counters or sequencers.

Dividers can be **synced** to a number of sources. This is an easy way to create a motif, or repeating phrase. For instance, techno from the early 90's like L.A. Style's "James Brown is Dead" often featured a 16 count phrase that played a note every two or three steps.. a perfect application for the divider.

The coloured [ ? ] control is the random seed for each divider. If you want a different pattern, drag the control vertically like any knob. It will change colour in response.



## Divider



examples of divider output

Divider and period counters are intended to be used to define the major structural components of the sequence, such as measure length, passage, et c. though of course they can both be applied as the user sees fit.

Both are built in to other sections of Hyperion's architecture: the period counters have dedicated buttons for resetting the lfes, and the dividers have dedicated buttons in the gate section. These are explained in the appropriate section.

Power users may wish to note that the **min** setting overrides **max** if the settings overlap.

Finding the right random seed value for a gating pattern can reserve the sequencers for other applications.



## LFOs

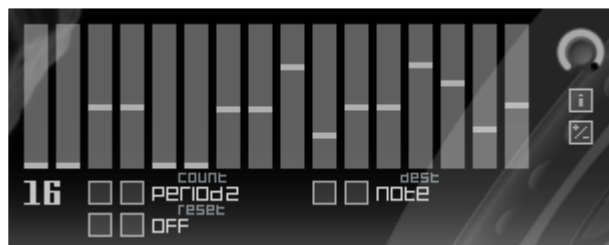
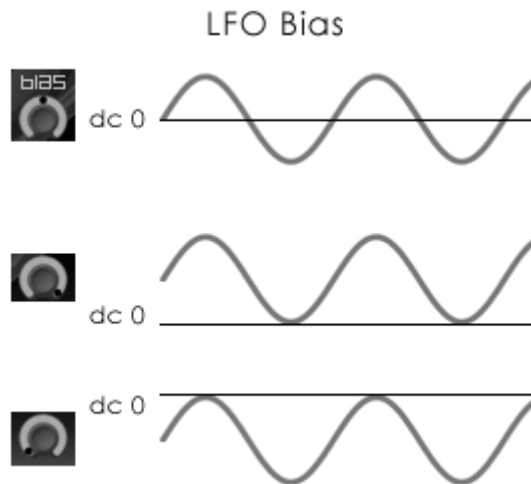
The LFOs are conventional, and include features appropriate for sequencing. The LFOs are **tempo synced**, meaning that if you change the tempo of a patch, the output should be similar if not exact.

Like the **period** counters, the LFOs can be **synced** to reset at a specified number of beats. Both LFOs can also be individually selected to be reset by either or both of the period counters.

LFOs offer sine, saw, ramp, square and triangular waveforms, which are selected by dragging vertically.

Normally, LFOs have a negative and positive polarity. The **bias** parameter shifts the center of the LFO so that the duty cycle can be entirely negative or entirely positive. This can be useful for finding the right setting when the LFO is used for triggering thresholds, like gates.

Setting the **rate** parameter higher than the clock rate can create "random," or cross-rhythm modulations you will hear in several of the presets. Changing the **phase** of the LFO will translate the modulation across the sequence.



## Sequences

The sequences are straightforward, and have assignable **count** and to **reset** sources. The sequence output is normally positive, **invert** and **+/-** options are located below the **amount** knob.

The third sequence can be assigned to modulate the counters and LFO 1. The first and second sequences are modulated by these sources.

When counters are assigned as the 'count' source for sequences 1 and 2, the sequences enter a unique mode where the counter value references the sequence step. Counter effects like bounce can then be used to create sequence modulations.



## Step modifier

The step modifier generates a random sequence that can be **reset** by a variety of sources. It has two assignable taps that can be **inverted**.

Like the **divider** counters, the random sequence is determined by a 'seed' value. Adjusting the [ ? ] control by dragging it vertically will change the output.

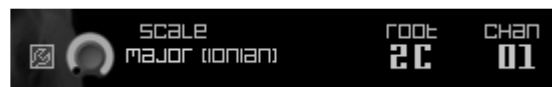
The counterpart to the step modifier is the **division** panel, named for continuity with previous VST, and not to be confused with the new **divider** counters.

The **division** panel splits each clock step into multitriggers. The buttons under each slider force multitriggers when they are held down, and the sliders set a percentage for each division to occur in the sequence. These probabilities are generated using the same random seed value as the **step mod** section.

The **2nd** option divides the step into two parts and only triggers the second. For instance, if Hyperion is set to generate 16th notes, this would create a 32nd note halfway through the step.

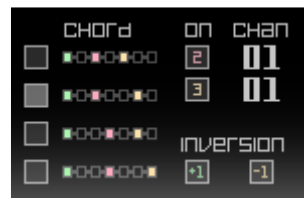
Pitch modulation of multitriggered notes is created using the **phase**, **rate** and **amt** settings. These control a triangle-shaped LFO which modulates the pitch of subsequent triggers. By adjusting these parameters, you should be able to shape almost any progression of notes for the multitrigger.

This LFO is reset on each step which causes the same pattern to be output on each multitrigger. Assigning an LFO to these parameters creates different variation.



## Key

The root and scale are indicated at the top of the gui. The **wrench** button holds the current **scale** and **root** settings while they are adjusted, for precise adjustments in live use.



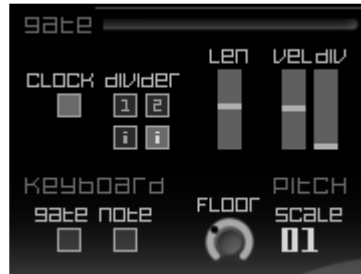
## Chord

Hyperion is oriented towards monophonic sequences, however a basic chord functionality is included.

The diagram illustrates which degrees in the scale are selected. The second and third notes in the chord (indicated by red and yellow colouration) can be assigned to separate MIDI channels for creative purposes.

The root of the chord may be raised by an octave, and the third note in the chord may be dropped by an octave. Note that if you are using a pentatonic (five note) or hexatonic (six note) scale, this may cause both to output the same MIDI note.





## Output

The **gate** and **pitch** sections indicate how the sequence is translated into MIDI notes.

## Gate

Three sources can be selected here for generating gate events: the **clock**, or either of the **dividers**, which can also be **inverted** to suppress gate events.

Other sources can also create or cancel gate events, so the output is derived from the sum of assigned modulators.

Sliders specify the **length** and **velocity** of the MIDI note. The velocity of multitriggers produced by the division section can be lowered using the **div** slider. This can create ghost, or grace note effects on multitriggers.

## Pitch

All pitch modulation is **scaled** from 1 to 4 octaves: when scale is set to 1, the range of the sequencers assigned to MIDI note is 1 octave, and the range of the LFO is 1 octave, although the sum of these modulators may exceed one octave.

If you are creating a patch with a dramatic note range, adjust the **amt** knobs for each source accordingly.

The note **floor** adds a -1 to +1 octave constant to note modulation that is not affected by the **scale** setting.

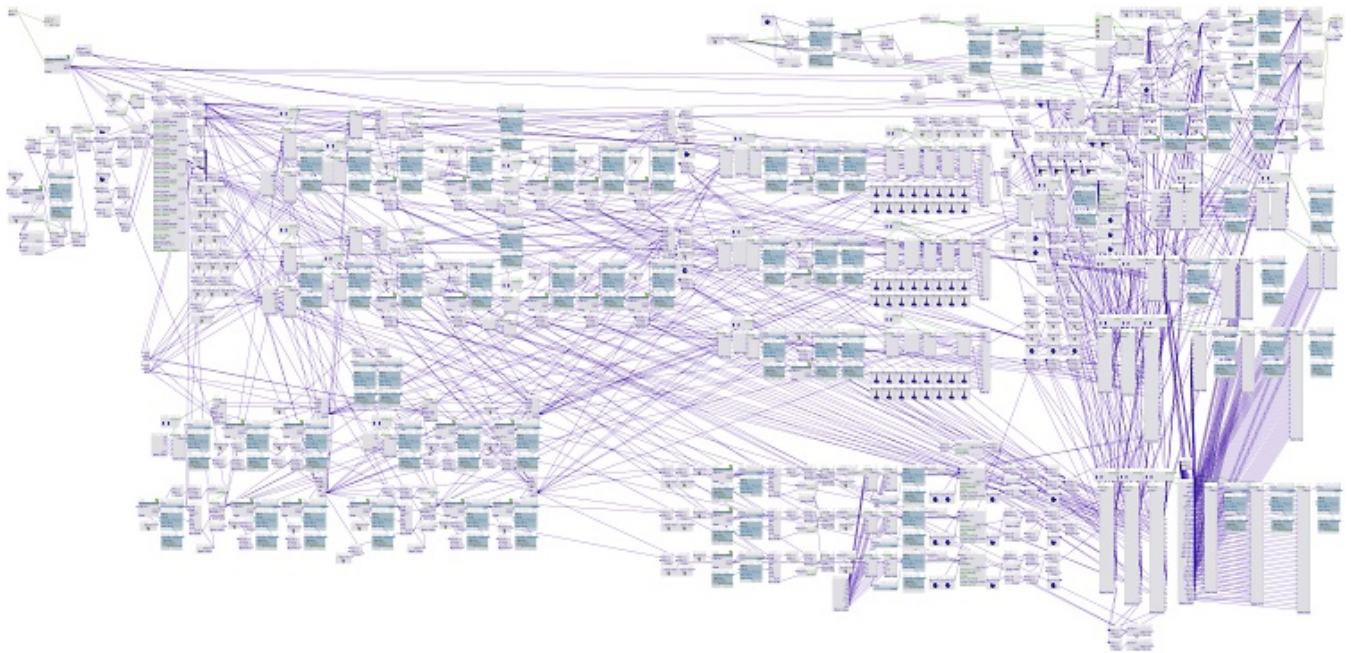
## Keyboard

Hyperion will transpose generated sequences in scale in response to MIDI input when **note** is selected. Transposition is based on the played note's relation to the **root** note indicated on the top central panel.

Similarly, the **gate** button elects for MIDI notes to only be output when a key is held.







## Count and reset

Most of the sources used for counting and resetting are explained above. Here is a complete list of the available sources with descriptions of items not explained previously:

Gate	Indicates a key press, eg. MIDI NoteOn event
Clock	
Ct 1 loop	Counters output a pulse on steps when they loop, wrap or bounce
Ct 2 loop	
Divider 1	
Divider 2	
Period 1	
Period 2	
LFO 2	Indicates the step after the biased LFO waveform makes a positive zero crossing
LFO 3	

Note that when counters are assigned as the 'count' source for sequences, the counter's value references the sequencer step. The counter loop pulse is available for resetting sequences 1 and 2 and for resetting the step modulator.

## List of modulation destinations

Here is the complete list of destinations with elaboration for items not explained previously:

Note	refers to pitch value of MIDI output
Length	length of MIDI note. 100% = full duration of clock pulse
Vel	velocity of MIDI note
Chord	pitch of 2nd and 3rd notes in chord. Does not adjust pitch of primary note output
Chord vel	velocity of 2nd and 3rd notes in chord. Does not adjust velocity of primary note output
Gate	modulates MIDI note off/on
Gate 2	note off/on for 2nd note in chord only
Gate 3	note off/on for 3rd note in chord only
Gate 2+3	note off/on for 2nd and 3rd note in chord

## List of modulation destinations (continued)

Ct # rpt	repeat coefficient of counter 1 or 2
Ct # inc	increment coefficient of counter 1 or 2
Ct # start	start coefficient of counter 1 or 2
Ct # low	low coefficient of counter 1 or 2
Ct # high	high coefficient of counter 1 or 2
Ct # amt	amount coefficient of counter 1 or 2

Note that the range of LFOs and sequences when applied to counter values translates as 0 to 10.

Seq # amt	amount coefficient of sequence 1, 2 or 3
Step mod #	amount of step mod tap 1 or 2
Step /2	% threshold of /2 division
Step /3	% threshold of /3 division
Step /4	% threshold of /4 division
Step /2nd	% threshold of /2nd division
Step ph	phase of triangle LFO used to modulate pitch of multitriggers
Step rate	rate of triangle LFO used to modulate pitch of multitriggers
Step amt	amount of triangle LFO used to modulate pitch of multitriggers
LFO # ph	phase of LFO 1 or 2
LFO # rate	rate of LFO 1 or 2
LFO # amt	amount of LFO 1 or 2

## Sequencing tips

As of this printing, Hyperion is only about a week old, about a month since I started developing it, and I'm still discovering modulations that produce unanticipated musical forms.

Hyperion **is** a flexible platform for generating a wide variety of MIDI sequences which can be dynamically modulated, it is **not** an instant music creator, as passages of music usually progress and resolve, and often resist strictly mechanistic approaches as being too predictable to hold the attention of the audience.

A period of experimentation with Hyperion will reveal that it is capable of producing surprising and interesting results, despite being a computation (only minor elements of Hyperion are generated 'randomly').

Hyperion has some utility for automating variation within an intended sequence. To me it seems more suitable as a vehicle for exploration (hence the UFO).

Here are a few techniques to keep in mind:

The counters allow you to apply basic mathematic patterns you are already familiar with, and which are commonly heard in music. Dynamic counter output is created the same way that more complex products are generated in math, by using prime or odd numbers that do not divide into each other evenly.

The examples illustrated on the counter mode diagram use 3 into 8 - in the 'wrap' mode, a descending pattern is created by the 'aliasing,' or 'moire effect' of 3 not dividing evenly into 8. Using 5 instead of 3 produces the same pattern in reverse.

If you aren't a 'math person,' you will notice that counter parameters go up to 16, so the operations are all within the range of integers we are familiar with.

Patterns don't always have to be complex: using 4 into 8, or 8 into 8 creates much simpler output. Switching modes between wrap and bounce produces different 'octave jumping' sequences which are often used in music.

You can make use of both counters by alternating or summing them, ie. using one counter with an octave jumping pattern and the other with a 3 in 8 pattern creates a mathematic sequence unobtainable with only one counter.

## Sequencing tips (continued)

The LFOs are also capable of creating complex output using only a few settings. Because the effect is not heard continuously as it is in a synthesizer, it helps to have a clear idea of how phase is modulated when syncing LFOs.

At the lowest phase setting, all LFOs are positioned at the positive zero crossing, as oscillators conventionally are. If a falling and rising effect is desired using a sine LFO, a 90 degree phase setting is appropriate, found 1/4 of the way through the dial. This corresponds to the peak of the sine waveform.

Being aware of phase and rate allows LFOs to be easily used to increase note events or velocity swells at a certain part of the measure. Using one LFO to amplify another can add further refinement.

The LFO rate settings are relative to the tempo. The highest portion of the rate dial allows a selection of rates that supercede 16th and 32nd notes and will create cross rhythms, again, like aliasing or the moire effect. These add an active, human "feel" to sequences.

Reserve sequence 3 for modulating the counters or LFO 1. If sequencer 3 is counting a group of notes (eg. a period counter,) modulating the counting increment, a boundary, or LFO rate adds a discernible sequence variation. The sequencer allows these variations to be set to occur ie. every 8th measure to increase tension.

The dividers are intended to rapidly split a sequence into shorter segments and can be applied to gate events or to pitch via other modulators. Syncing them to a discrete interval (like a measure of half measure) can create rhythmic signatures that will add drive to a sequence.

Both dividers and period counters have two instances, which are situated next to each other in the assign lists for counting and resetting. Toggling between period 1 and period 2 is an easy way of creating a variation in performance.

Dividers always output a pulse on the step they are reset on. Using one (or both) dividers to create gate events will add an aggressive feel to the sequence by starting on the first step. Using the invert buttons to subtract the dividers from the gate will omit the first step.

The step modifiers output a random sequence determined by the seed value, like the dividers. Finding a fortuitous seed value for a patch can output a desired sequence and reserve sequencers for other tasks.

Assigning one step mod to gate 2 and the other to gate 3 will play one or both of the chord notes on each step. Assigning a step mod to 'chord' will increase the pitch of chord notes so that their sequence is not strictly correlated to the primary output.

Some modulations produce interesting effects that may not seem obvious. If you are modulating MIDI note with one source on every step to create a pattern, and applying another modulator to transpose the pattern every 16 steps, applying a third modulator to the per-step pattern will create riffs that are oriented and expand around the root of a chord.

## License

Hyperion VST may not be redistributed or resold. There is no expression of guarantee due to the use of third party resources.

