

SCORE GENERATING LINDENMAYER SYSTEMS IN THE GENERALIZED CONTEXTUAL GROUP

Michael Gogins

Irreducible Productions
New York, New York, United States
michael.gogins@gmail.com

ABSTRACT

Lindenmayer systems have long been used in computer music as score generators. I demonstrate a context-free Lindenmayer system based on a group of operations, the Generalized Contextual Group (GCG), which abstracts and extends neo-Riemannian transformations. This Lindenmayer system is given commands representing operations from the GCG, as well as other operations on dimensions and quotients of chord space; these commands move a turtle, which represents several simultaneous voices of music, from one chord to another. A sequence of chords thus generated defines a score.

1. INTRODUCTION

Algorithmic score generators have been based on many different generative systems (see [1] for a survey, [2] for an up to date list). Lindenmayer systems [3] have been among the most widely used of such systems for composition (e.g. [4, 5, 6, 7]). Usually a graphics turtle controlled by the Lindenmayer system moves about in a space with time and pitch as coordinates, in essence drawing notes on a piano roll score.

Some systems of this type can, in principle, generate any possible score. Yet the approach tends to abstract from all pragmatic music theory, i.e. all rules that define whether a piece of music is in some sense “well-formed” — or, to avoid possible misinterpretations of this term, in which it is easier for the audience to hear what is going on. Thus, as with so many generative algorithms, the search space for pieces of music that might be good to hear becomes vast and full of noise, blocking composition. There are at least two potential solutions: sample and map the search space such that listenable pieces are close together, for which purpose generated scores must depend continuously upon numerical parameters in the score generator [8]; or base the system upon a set of musical operations that are themselves “well-formed,” such that productions of the system are simply more likely to be listenable. This approach is of course quite common. But it is critical to identify the most suitable operations, and to implement them in the most usable way.

Obstacles and issues are various and profound. As Lerdaahl [9] notes, the grammar used by the composer may be parasitic on the practices of the past, and thus fail to be capable of originality; or it may fail to produce music that can be parsed by the grammar used by the audience, who will thus fail to hear the music. I believe it is essential to select generative operations that are both abstract enough to transcend the past, and sufficiently fundamental to ensure a degree of listenability.

My general strategy is to adopt or adapt ideas from mathematical music theory that seem not only suited to score generation, but also related to traditional training in pragmatic music theory; to decompose these ideas into logically independent, or atomic, operations; to implement these operations using group representations; and finally to generate sequences of these operations using some recursive system (such as Lindenmayer systems). Each group representation must be *complete* — i.e. capable of generating every heard element of the group, even if that means incorporating information that is *not* actually heard into the elements of the group.

Most music theory has not been mathematically precise or unambiguous enough to serve as a direct source of group operations for algorithmic score generation. Recently, however, mathematical musicologists have begun producing theories that are indeed precise and unambiguous. There are a variety of such theories. One of the first was Lerdaahl and Jackendoff’s generative grammar [10], motivated by Lerdaahl’s remarks above; it does not, however, seem to have fathered a fertile line of score generators. Perhaps this is because the theory still contains ambiguities.

The present work is based on transformations that move one point in chord space, to another point in chord space, using operations from Fiore and Satyendra’s Generalized Contextual Group [11] — an extension of the neo-Riemannian transformations of Lewin [12]. These transformations are more limited in scope than [10], but have the advantage of being both complete and unambiguous. Operations from the Generalized Contextual Group tend to produce normative voice-leading, and these operations can easily be composed to produce functional chord progressions.

2. DEFINITIONS

Musicians and musicologists often tacitly make one set of assumptions for an object in one context, yet a different set of assumptions for the same object in a different context; e.g. a $C7$ chord could be assumed under range equivalence in one context, so that its voicings can be considered; yet the same chord could be assumed under octave equivalence in another context, so that its voicings need *not* be considered. Usually the implied equivalences are clear from the context — but not always. We shall need to be much more precise in order to derive complete group operations for score generation. The following definitions are used.

Metric A measure of distance in some space.

Equivalence class Defines elements of a set as identical iff they are identical in one or more properties.

Pitch The sensation that we perceive from the frequency of a tone. Middle C is 60, $C\sharp$ above middle C is 61, 12 is the size of the octave, and so on; but pitches can just as easily be fractions or real numbers.

Pitch-class Pitch under octave equivalence, e.g. C_4 and C_5 are identical and can be labelled simply C . Pitch-classes are numbers, not necessarily integers, in \mathbb{R}_{12} ; 0 is the pitch-class C , 0.5 is the pitch-class one quarter tone above C .

Transposition A function from pitch-classes onto pitch-classes, $T_n(X) := x + n$ in \mathbb{R}_{12} .

Inversion Mathematically, *reflection*: a function from pitch-classes onto pitch-classes, $I_n(X) := -x + n$ in \mathbb{R}_{12} . *Not* the same as the musician’s colloquial term “inversion,” which denotes the permutation of octaves in the pitches of a chord.

Chord A set of perceptually distinct pitches sounding at the same time.

Score For the purposes of the present work, a sequence of more or less fleeting chords.

Chord space A Euclidean space with one dimension of pitch for each voice of a chord. Thus a chord can also be considered to be a point in chord space, e.g. a triad is a point in \mathbb{R}^3 , a seventh chord is a point in \mathbb{R}^4 , and so on.

Voice-leading An operation that sends one chord to another by moving voices, e.g. $\{C_4, E_4, G_4\} \rightarrow \{C_4, F_4, A_4\}$.

Quotient space Also *orbifold*. A space that is produced by gluing together points that have the same equivalence class. Callender, Quinn, and Tymoczko [13] consider the following equivalence classes for chords:

O *Octave equivalence*: a chord is an ordered set of pitch-classes or *pitch-class segment*, written with parentheses, e.g. (C, E, G) is the same chord as (C_3, E_3, G_3) or (C_5, E_2, G_5) .

P *Permutational equivalence*: two chords with the same pitches but whose voices occur in different order are the same, written with braces, e.g. $\{C, E, G\}$ is the same chord as $\{G, E, C\}$.

T *Transpositional equivalence*: a chord is a set of interval classes, e.g. $CM9$ is the same chord as $F\sharp M9$.

I *Inversional equivalence*: the voices of a chord may reflect around a center, e.g. (C, E, G) inverted by $F\sharp$ is the same as $(F\sharp, D, B)$.

C *Cardinality equivalence*: chords having the same pitches are the same even if each has a different number of voices per pitch, e.g. $\{C, E, G\}$ is the same chord as $\{C, E, G, C\}$.

R *Range equivalence*: in my own work, I have found it useful to consider a chord as a set of pitches under the equivalence of some multiple of the octave.

Each *product* of equivalence classes also defines a quotient space. Short moves in these orbifolds tend to be musically normative and, in particular, to correspond with smooth voice-leading. Tymoczko [14] and Callender, Quinn, and Tymoczko [13] consider a wide variety of more complex and musically meaningful equivalence classes, of which some of the most important are:

OPC A point is a *pitch-class set*, the most common musical meaning of chord: pitches without respect to octave, without respect to order, and without respect to number of voices; written with braces, e.g. $\{C, E, G\}$.

OP The orbifold consisting of chords without respect to octave and without respect to order, e.g. the orbifold for triads in Tymoczko [14] and [15], which is a prism with augmented triads down the middle, surrounded by columns of major and minor triads, and unisons along one edge; the orbifold is defined by twisting the prism 1/3 turn and gluing its ends together.

OPTI Also *set class* or *chord type*. The same as **OP**, but without respect to transposition or inversion (e.g., for 7th chords, contains 1 major 7th, 1 minor 7th, 1 dominant or half-diminished 7th, and 1 diminished 7th in the middle). **OPTI** is the P space in the `CsoundAC.Voiceleading` algorithmic composition node. Voice-leadings in **OPTI** are movements from one type of chord to another.

PR The orbifold of chords having the same cardinality, unordered, distinguishing octaves within a range of pitches. For triads it is a prism containing 3 columns with augmented triads down the middle, each surrounded by columns of major and minor triads (but all in the same inversion), and unisons along one edge; again the orbifold is defined by twisting the prism 1/3 turn and gluing its ends together. Voice-leading in **PR** are chord progressions that take account of octaves, i.e. voice-leading in the musician’s colloquial sense of the term.

3. IMPLEMENTATION

3.1. Primitive Operations

The original neo-Riemannian transformations are those that, to paraphrase Cohn [16], map a major triad to a minor triad (or vice versa), and that applied once more map the resulting chord back to the original chord (i.e. are involutions):

P Parallel: transform a major triad to the minor triad with the same root, or *vice versa*.

L Leading-tone exchange (or Leittonwechsel): transform a major triad to the minor triad one major third higher, or *vice versa*.

R Relative: transform a major triad to its relative minor triad, or *vice versa*.

These transformations are called *contextual inversions* because the voice-leading depends upon the modal context, i.e. the voices move one way from a major chord, and another way from a minor chord. *P* and *L* alone can generate the entire group *M* of the 12 major and 12 minor triads. *P*, *L*, and *R* also generate *M* — in the form of the *Tonnetz* of Oettingen and Euler.

Hook [17] and Childs [18] extend neo-Riemannian transformations to chords that are not necessarily major or minor, and that do not necessarily have 3 voices. Fiore and Satyendra [11] further extend these transformations to define the Generalized Contextual Group, by abstracting the contextual shift between major and minor to *any* contextual inversion. However, the chords operated on by the group must be *pitch-class segments*, i.e. chords in **O**, not the more familiar *pitch-class sets* in **OP**. The set of all chords generated by the GCG, M_S , can be generated from any pitch-class segment *S* through the following operations:

K Interchange by inversion: Given a pitch-class segment $S = (s_1, s_2, s_3 \dots s_n)$,

$$K(S) := I_{s_1+s_2}(S) \quad (1)$$

I.e., *K* is that unique inversion of *S* which interchanges its first 2 pitch-classes. All pitch-class segments containing a non-tritone interval can be inverted in this way.

Q_i *Contextual transposition:* Given a pitch-class segment *X*,

$$Q_i(X) := \begin{cases} T_i(X) & \text{if } X \text{ is a } T\text{-form of } S \\ T_{-i}(X) & \text{if } X \text{ is an } I\text{-form of } S \end{cases} \quad (2)$$

N.B.: If *S* is asymmetrical, then *K* and Q_i induces K' and Q'_i on the *unordered* form of *S*. That is, the Generalized Contextual Group generates M_S for *all* pitch-class segments, and it generates M_S for all *inversionally asymmetrical* pitch-class sets.

There is no claim here that these transformations are the only or best source of a compositional vocabulary, nor that they are in any extensive sense musically complete, i.e. that they can generate all possible scores. The present work does, however, demonstrate some of the consequences of basing a score generator on “well-formed” operations that make musical sense in terms of voice leading. I hope that these operations also are sufficiently abstract to enable the generation of music that could not easily have been produced in the past. For example, these transformations are not limited to the Western system of 12-tone equal temperament, or indeed of tonality in any sense, but apply to any system of temperament, any scale, and any chords containing at least one interval that is not a tritone.

3.2. Lindenmayer System

A deterministic, context-free Lindenmayer system [3] consists of a tuple $\langle V, \omega, P, n \rangle$, where

V is an alphabet of letters; some letters are commands for instructing a turtle how to move about in some sort of space and draw some sort of figure, other letters are not commands but rather are placeholders or symbols;

ω is an axiom, a non-empty sequence of words in *V*;

P is a non-empty set of productions. A production $(a, \chi) \in P$ specifies that the letter *a* is to be replaced by the word χ ; and

n is an integer specifying how many times the productions are to be applied to the axiom.

When the productions are iteratively applied to the axiom, each letter *a* in the perhaps rather short axiom is replaced by the corresponding word χ , so that the axiom is expanded into a perhaps very long string of letters. The turtle then interprets these letters as a string of commands, and thus generates a score. Because iteratively applying the productions to the axiom is a *recursive procedure*, Lindenmayer systems are actually very powerful.

The Lindenmayer system used here has a turtle that represents a musical chord as a point in $\mathbf{R} \times \mathbf{D} \times \mathbf{L}$, that is, a set of ordered pitches that distinguishes octaves within some range equivalence, plus duration \mathbf{D} , plus loudness \mathbf{L} . The \mathbf{R} space, in turn, is decomposed into \mathbf{O} and \mathbf{V} ; this latter is the additive group defined by the zero-based index of the octave-distinguished permutations of the \mathbf{O} chord within \mathbf{R} . This index wraps around as necessary. We do not consider the timbre of the voices, except that timbres may be arbitrarily assigned to the voices. So that the type of chord operated on by the Generalized Contextual Group can be changed in the system, operations are also provided for operating independently in \mathbf{P} (change of chord type) and \mathbf{T} (transposition). To make them easier to remember, most of the commands actually consist of more than one letter. Table 1 summarizes the turtle commands and their meanings, with reference to the particular space in which the command operates.

Note that there is no command for moving the turtle back and forth in time. Time is defined by the mere sequence of chords in the final production of the system. If two identical chords are generated in sequence, they are tied to produce a single chord. Similarly, if two chords in sequence share one or more tones, those tones are tied. Rhythms can be generated either by generating shorter or longer sequences of identical chords, or by incrementing and decrementing the durations of chords.

4. EXAMPLES

As a first example of using this Lindenmayer system, let us reproduce the chord progression from Beethoven's Symphony No. 9, 2nd movement, which is a type example of neo-Riemannian progressions [19], $C, a, F, d, B\flat, g, E\flat, c, A\flat, f, D\flat, b\flat, G\flat, e\flat, B, g\sharp, E, c\sharp, A$. It is generated by starting with C and then applying the sequence RL 9 times.

As noted previously, the neo-Reimannian PLR group can be generated by the Generalized Contextual Group for triads: $P(X)$ is $Q_9(K(X))$, $L(X)$ is $K(Q_7(X))$, and $R(X)$ is $K(X)$. Therefore, the same progression can be generated by starting with C , then applying $K(X) K(Q_7(X))$ 9 times. The Lindenmayer system for doing this is shown in Equation 3, and generates the score shown in Figure 1:

$$\begin{aligned} \omega &:= S(0,4,7) a WV \\ P_1 &:= a \leftarrow K WV Q7 K WV a \\ n &:= 9 \end{aligned} \quad (3)$$

The use of the Lindenmayer system to quickly and easily generate variations on such sequences can now be demonstrated. To begin with we can substitute $Cm9$ for the starting C chord, as shown in Equation 4 and Figure 2:

$$\begin{aligned} \omega &:= S(0,3,7,10,14) a WV \\ P_1 &:= a \leftarrow K WV Q7 K WV a \\ n &:= 9 \end{aligned} \quad (4)$$

Table 1. Turtle Commands

Command	Space	Operation
[Push the current turtle state onto a stack.
]		Pop the current turtle state from the stack.
$D * n$	D	Multiply the duration of the turtle chord by n (default 2).
D / n	D	Divide the duration of the turtle chord by n (default 2).
$Iv=i$		Assign instrument i to voice v .
K	OC	Apply inversion by interchange to the turtle chord.
$L * n$	L	Multiply the loudness of the turtle chord by n (default 2).
L / n	L	Divide the loudness of the turtle chord by n (default 2).
$P(p_1, p_2, \dots, p_n)$	OPTI	Set the chord type to the indicated set class.
P_{chord}	OPTI	Set the chord type to the jazz-named chord type.
Qn	OC	Apply contextual transposition to the turtle chord.
Rn	R	Set the size of the range.
$S(p_1, p_2, \dots, p_n)$	R	Set the turtle chord to the indicated pitch-class segment.
$T+n$	O	Transpose the turtle chord up by n semitones.
$T-n$	O	Transpose the turtle chord down by n semitones.
$V+n$	V	Increment voicing, i.e. add n octaves to the permutation of the turtle chord in R .
$V-n$	V	Decrement voicing, i.e. subtract n octaves from the permutation of the turtle chord in R .
wCn	OCV	Write the turtle chord n times into the score.
wVn	OCR	Write the turtle chord n times into the score using the voicing in R closest to the previous chord.



Figure 1. R L Sequence starting from C



Figure 2. R L Sequence starting from Cm9

Then we can add additional Lindenmayer commands to arpeggiate these chords (Equation 5, Figure 3). The close voice-leading produce various melodies.

$$\begin{aligned}
 \omega &:= S(0, 3, 7, 10, 14) R36 D/4.0 WC a \\
 P_1 &:= a \leftarrow K b Q7 K b a \\
 P_2 &:= b \leftarrow WC V + 7 WC V - 5 b \\
 n &:= 8
 \end{aligned}
 \tag{5}$$

The final example shows an excerpt from the score of a finished composition, generated in one pass by the Lindenmayer system (Figure 4).

5. FUTURE DIRECTIONS

I regard the present work as a starting point. Future enhancements and extensions could include:

- Ensuring that the implementation of the GCG operations actually does work in arbitrary tuning systems.
- Enabling the system to vary the number of voices during score generation.
- Allowing the operations to apply to arbitrary sets of zero or more notes, instead of automatically generating a single chord; that is, the operations would conform notes generated within a specific time span (by another process or processes) to the chord and voicing generated by the GCG operation.
- Allowing each voice to independently carry its own loudness, duration, and other properties.

6. REFERENCES

- [1] Gerhard Nierhaus. *Algorithmic Composition: Paradigms of Automated Music Generation*. Springer-WienNewYork, Vienna, 2009.
- [2] Christopher Ariza. Algorithmic.net: A Lexicon of Systems and Research in Computer Music in Computer Aided Music Composition. <http://www.flexatone.net/algonet>.
- [3] Przemyslaw Prusinkiewicz and Artistic Lindenmayer. *The Algorithmic Beauty of Plants*. Springer-Verlag, New York, 1996 [1991]. Available online at <http://algorithmicbotany.org/papers>.
- [4] P. Prusinkiewicz. Score generation with L-systems. In *Proceedings of the 1986 International Computer Music Conference*, pages 455–457, 1986.
- [5] Michael Gogins. Fractal Music with String Rewriting Grammars. *News of Music*, 13:146–170, Winter 1992.
- [6] R.L. DuBois. *Applications of Generative String-Substitution Systems in Computer Music*. PhD thesis, Columbia University, 2003.
- [7] S. Manousakis. Musical l-systems. *Master's thesis, Conservatoire Royal de La Haye*, 2006.
- [8] Michael Gogins. ...How I Became Obsessed with Finding a Mandelbrot Set for Sounds. *News of Music*, 13:129–139, Winter 1992.
- [9] F. Lerdahl. Cognitive constraints on compositional systems. *Contemporary Music Review*, 6(2):97–121, 1992.

The image displays a musical score for an arpeggiated Cm9 chord sequence. The score is written in 4/4 time and consists of five systems of two staves each (treble and bass clef). The first system begins with a measure number '1' and a key signature of one flat (Bb). The melody in the treble clef starts with a quarter rest, followed by a sequence of eighth and quarter notes: Bb, C, D, Eb, F, G, Ab, Bb. The bass clef accompaniment consists of a steady eighth-note arpeggiated pattern: Bb, C, D, Eb, F, G, Ab, Bb. The second system starts at measure 4, the third at measure 7, the fourth at measure 10, and the fifth at measure 13. The sequence concludes with a double bar line at the end of the fifth system.

Figure 3. R L Sequence from *Cm9*, arpeggiated

56

Instrument 1

Instrument 2

Instrument 3

Instrument 4

Instrument 3

Instrument 4

Instrument 3

Instrument 4

Figure 4. Extract from *mkg-2009-02-03-b*

- [10] F. Lerdahl and R. Jackendoff. *A Generative Theory of Tonal Music*. MIT Press, 1983.
- [11] T.M. Fiore and R. Satyendra. Generalized Contextual Groups. *Music Theory Online*, 11(3), 2005.
- [12] D. Lewin. A Formal Theory of Generalized Tonal Functions. *Journal of Music Theory*, 26(1):32–60, 1982.
- [13] Clifton Callender, Ian Quinn, and Dmitri Tymoczko. Generalized voice-leading spaces. *Science*, 320:346–348, 2008.
- [14] Dmitri Tymoczko. The Geometry of Musical Chords. *Science*, 313:72–74, 2006.
- [15] Michael Gogins. Score generation in voice-leading and chord spaces. In Georg Essl and Ichiro Fujinaga, editors, *Proceedings of the 2006 International Computer Music Conference*, San Francisco, California, 2006. International Computer Music Association.
- [16] R. Cohn. Neo-Riemannian Operations, Parsimonious Trichords, and their Tonnetz Representations. *Journal of Music Theory*, 41(1):1–66, 1997.
- [17] J. Hook. Uniform Triadic Transformations. *Journal of Music Theory*, 46(1-2):57, 2002.
- [18] A. Childs. Moving Beyond Neo-Riemannian Triads: Exploring a Transformational Model for Seventh Chords. *Journal of Music Theory*, 42(2):181–193, 1998.
- [19] Richard Cohn. Properties and generability of transpositionally invariant sets. *Journal of Music Theory*, 35(1-2):1–32, 1991.