

A Few More Words About James Tenney: Dissonant Counterpoint and Statistical Feedback

Larry Polansky*, Alex Barnett*, and Michael Winter†

* Dartmouth College

† University of California Santa Barbara

November 6, 2010

Abstract

This paper discusses a compositional algorithm, important in many of the works of James Tenney, which models a melodic principle known as *dissonant counterpoint*. The technique synthesizes two apparently disparate musical ideas—dissonant counterpoint and statistical feedback—and has a broad range of applications to music which employs non-deterministic (i.e. randomized) methods. First, we describe the historical context of Tenney’s interest in dissonant counterpoint, noting its connection to composer/theorist Charles Ames’ ideas of statistical feedback in computer-aided composition. Next, we describe the algorithm in both intuitive and mathematical terms, and analyze its behavior and limiting cases via numerical simulations and rigorous proof. Finally, we describe specific examples and generalizations used in Tenney’s music, and provide simple computer code for further experimentation.

Contents

1	Tenney, Dissonant Counterpoint, and Statistical Feedback	2
1.1	Tenney and Dissonant Counterpoint	2
1.2	Statistical Feedback: Probability vs. statistics	4
2	The Dissonant Counterpoint Algorithm	7
2.1	Informal description	7
2.2	Formal description	9
2.3	Curvature of the growth function	14
2.4	Vanishing power: “max-1” version and the effect of weights .	16

3	Examples from Tenney’s Work	19
3.1	<i>Seegersongs</i>	21
3.2	The <i>Spectrum</i> pieces	22
3.3	<i>To Weave (a meditation)</i>	25
3.4	<i>Panacousticon</i>	25
3.5	<i>Arbor Vitae</i>	26
4	Conclusion	28
A	Matlab/Octave code examples	29

1 Tenney, Dissonant Counterpoint, and Statistical Feedback

“Carl Ruggles has developed a process for himself in writing melodies for polyphonic purposes which embodies a new principle and is more purely contrapuntal than a consideration of harmonic intervals. He finds that if the same note is repeated in a melody before enough notes have intervened to remove the impression of the original note, there is a sense of tautology, because the melody should have proceeded to a fresh note instead of to a note already in the consciousness of the listener. Therefore Ruggles writes at least seven or eight different notes in a melody before allowing himself to repeat the same note, even in the octave.”

Henry Cowell, [13] (pp. 41–42)

“Avoid repetition of any tone until at least six progressions have been made.”

Charles Seeger, “*Manual of Dissonant Counterpoint*” [26] (p. 174)

1.1 Tenney and Dissonant Counterpoint

The music of James Tenney often invokes an asynchronous musical community of collaborators past and present. Many of his pieces are dedicated to other composers, and poetically reimagine their ideas. In these works Tenney sometimes expresses connections to another composer’s music via sophisticated transformations of the dedicatee’s compositional methods. For

example *Bridge*¹ (1984), for two pianos, attempts to resolve the musical and theoretical connections between two composers—Partch and Cage—both of whom influenced Tenney [35, 33]. The resolution, or “reconciliation” [35], in this case the marriage of Cage-influenced chance procedures and Partch-influenced extended rational tunings, is the piece itself. The construction of that particular bridge made possible, for Tenney, a new style with interesting genealogies.

As a composer, performer, and teacher, Tenney took musical genealogy seriously. He seldom published technical descriptions of his own work,² but he wrote several innovative theoretical essays on the work of others (see for example [33, 32]). Some of these explications are, in retrospect, transparent theoretical conduits to his own ideas. Many of the historical connections in Tenney’s work (to the Seegers, Partch, Cage, Varèse, even Wolpe) appear in or are suggested by his titles. However, Tenney’s formal transformations of other composers’ ideas are less well understood. The few cases in which he wrote about his own pieces (for example, see [34]) demonstrate the amount of compositional planning that went into each composition.

In Tenney’s theoretical essay on the chronological evolution of Ruggles’ use of dissonance [31], he used simple statistical methods to explain Ruggles’ (and by extension, the Seegers’) melodic style. For example, he examined how long it took, on average, for specific pitch-classes and intervals to be repeated in a single melody. Tenney visualized these statistics in an unusual way: as sets of functions over (chronological) time—years, not measures. The article consists, for the most part, of graphs illustrating the statistical evolution of Ruggles’ atonality along various axes. In general, the x -axis is Ruggles’ compositional life itself. Tenney considered the *data*, such as the ways in which the lengths of unrepeated pitch-classes increased over time (and if in fact they did). Focusing on this specific aspect of Ruggles’ work allowed Tenney to explain to himself, in part, what was going on in the music.

In the two decades that followed that paper, Tenney widened his theoretical focus to include the music and ideas of what might be called the 1930s “American atonal school,” including Ruggles, Henry Cowell, Charles and Ruth Crawford Seeger, and others. These American composers em-

¹All Tenney scores referenced in this article are available from Smith Publications or Frog Peak Music. Most of the works mentioned are recorded commercially as well.

²Tenney’s personal papers, however, include many descriptions and notes pertaining to his pieces. Most of his music after about 1980 was written with the aid of a computer (after a long hiatus in that respect), and the software itself is extant. In addition, Tenney frequently described his compositional methods to his students.

ployed their own pre-compositional principles distinct from, but as rigorous as, European 12-tone composers of the same period. In the article on Ruggles, Tenney had described a way to statistically model an aspect of these composers' compositional intuitions. In the 1980s, he began to formally and computationally integrate Seeger's influential ideas of dissonant counterpoint [26, 27] into his own music. What seems at first to be a series of titular homages (in pieces like the *Seegersongs*, *Diaphonic Studies*, *To Weave (a meditation)*, and others) are in fact a complex, computer-based transplantation of dissonant counterpoint into the fertile soil of his own aesthetic.

In each of these pieces, and several others, Tenney employed a probabilistic technique which we call the *dissonant counterpoint algorithm*. Seegerian dissonant counterpoint encompassed a wide range of musical parameters (rhythm, tonality, intervallic use, meter, even form and orchestration). In this paper, we focus on an algorithm Tenney devised to make a certain kind of probabilistic selection (mostly pitch, but sometimes other things as well). This algorithm which was in part motivated by Tenney's interest in the ideas of dissonant counterpoint. To our knowledge he never published more than a cursory description of this technique [34]. One of the goals of our work is to present the algorithm and explore some of its features in a mathematical framework.

1.2 Statistical Feedback: Probability vs. statistics

“Along with backtracking, statistical feedback is probably the most pervasive technique used by my composing programs. As contrasted with random procedures which seek to create unpredictability or lack of pattern, statistical feedback actively seeks to bring a population of elements into conformity with a prescribed distribution. The basic trick is to maintain statistics describing how much each option has been used in the past and to bias the decisions in favor of those options which currently fall farthest short of their ideal representation.”

Charles Ames [3]

The dissonant counterpoint algorithm is a special case of what the composer and theorist Charles Ames calls *statistical feedback*:³ the current out-

³For a good explanation of this see [5]. However, several of Ames' other articles discuss it as well, including [8, 7, 6, 4, 2, 1].

come depends in some non-deterministic way upon previous outcomes.⁴ Tenney’s algorithm is an elegant, compositionally-motivated solution to this significant, if subtle compositional idea. Statistical feedback is another form of reconciliation—that of compositional method with musical results—and it has ramifications for any kind of computer- or anything-else-aided-composition or art form. We first give a general (and mathematically simple) introduction to this idea.

Imagine that we flip an unbiased coin $N = 1000$ times. We might end up with 579 heads and 421 tails. This is close to the equal statistical mean we might expect, want, or intend. With $N = 10000$ flips, we would do better, in the sense that although the numbers of heads or tails will most likely have larger differences from their expectation of 5000 than before, the *fraction* of heads is likely to be closer to $1/2$ than before. This illustrates the law of large numbers: the average of N outcomes from an independent and identically distributed (iid) random process converges (almost surely) to its expected value as N increases. For example, baseball wisdom holds that “any given team on any given day can beat any other given team.” But because the number of games played in a season, 162, is a pretty large number of trials, things generally (but not always) work out well for better teams.

But what about a more local observation of a small number of trials, or frame? For example, some run of ten flips might yield:

HTHHHHHTHT

This statistical frame contains, not surprisingly, something worse: seven *H*s, three *T*s.⁵ Nothing in our method suggests that we want that: the act of

⁴Ames [5], in discussing his early compositional use of this technique, describes it more simply as “the trick of maintaining statistics detailing how much each option has been used in the past, and of instituting decision-making processes which most greatly favor those options whose statistics fall farthest behind their intended distribution.” Also see [12], which surveys Ames’ work (up until 1992), and contains an alternate mathematical formalization of statistical feedback (p. 35).

⁵And, from a more local, musical perspective, there are five *H*s in a row, an example of what Ames refers to as “heterogeneity” or “dispersion” [5], and Polansky refers to as “clumping” [25, 21]. This is a slightly different, though related, problem to that of monitoring the global statistics of a probability distribution, but the solution also may employ statistical feedback. These terms refer to the difference between the following two, equally well-distributed coin toss statistics:

HHHHHTTTTT and *HTHTHTHTHT*

(pitch classes in one octave)



Figure 1: A simple melody generated by the algorithm, using a linear growth function (see next section), with $n = 12$ pitch elements lying in one octave. (Accidentals carry through the measure unless explicitly canceled.)

flipping an unbiased coin most likely (but not unequivocally) suggests that we desire a uniform distribution of outcomes. The random process creates a disjunct between compositional intention and statistical outcome.

Composers have long used probability distributions, but have not often worried about the conformance of observed statistics to probabilistic composition method over short time frames, what Ames calls “balance.”⁶ This is perhaps due to the typically small populations used in a piece of music, or because of a greater focus on method itself. Ames’ work suggests a variety of ways to gain compositional control over this relationship. Statistical feedback “colors” element probabilities so that over shorter time frames, the statistics (results) more closely correspond to the specified probabilities. A scientist might call this *variance reduction*; we will analyze this in Section 2.3.

Let’s return to the ten coin flips. We had seven H s, three T s. Using statistical feedback we can compensate so that our frame, of, say, twenty trials, is statistically better. The obvious thing to do is positively bias the probabilities of depauperate selections. For instance, we might now use for the eleventh toss $p(H) = .3$ and $p(T) = .7$, favoring the selection of a T . To paraphrase Ames, we use the preceding statistics as an input to the generating probability function.

⁶See, for example [8, 7, 5].

2 The Dissonant Counterpoint Algorithm

2.1 Informal description

Tenney often discussed his interest in “models,” and his criteria were in how the brain, the ear, the human accomplished something. His theories and compositional algorithms generally placed a high priority on the success of the theory or algorithm in elucidating some cognitive process.

The algorithm described here reflects this design goal in an efficient and elegant way. A list of n values is maintained, one for each pitch element, which will be interpreted as relative selection probabilities. We initialize these to be all equal, then proceed as follows:

1. Select one element from the list randomly, using the values as relative probabilities for choosing each element
2. Set the selected element’s value to zero
3. Increase the values of all the other elements in some deterministic way, for instance by adding a constant
4. Repeat (go back to step 1)

The algorithm is deceptively simple. Note that once selected, an element is temporarily removed from contention (its probability is zero). That element and all other unselected elements become more likely to be picked (their probabilities climb) on successive trials or ‘time steps’⁷ of the algorithm. The longer an element is *not* picked, the more likely it is that it *will* be picked.⁸ Tenney’s use of this algorithm is an extension and abstraction of one particular aspect of the compositional technique of Ruggles and/or Crawford Seeger, that of non-deterministic non-repetition of pitch-class or interval-class. Fig. 1 provides the simplest possible example: only pitch-classes are chosen by the algorithm. There is no explicit control of intervallic distribution (which would, of course, be of concern to Ruggles or Crawford Seeger). We give a more complex musical composition using this basic algorithm in Fig. 2.

⁷We use ‘time steps’ to refer to repetitions of the algorithm; note that this does not necessarily imply regular time intervals in a musical sense.

⁸Note that to fully “avoid repetitions of any tone until at least six progressions have been made” (as suggested by Seeger), an element’s probability would have to remain at zero for six trials after its choice. In Section 2.2, we show how this idea is incorporated in the general description of the algorithm, using growth functions with a high power to produce a similar result.

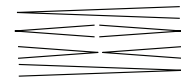
Demonstration 1

polansky
9/09

$\text{♩} = 120$ (or faster) (each measure = 4/4)

for three bass instruments
space = time, notes held to any length, articulated in any way

each player should impose their
own dynamic curve, for example:



using any dynamics
for the extremes.

Figure 2: A short example piece, a trio for any three bass instruments. Each instrument plays one of four pitch-classes, which are selected by the linear version of the dissonant counterpoint algorithm. Durations are chosen by a narrow Gaussian function whose varying mean follows a curve which begins quickly, gets slower, and then speeds up again. A simple set of stochastic functions determine the likelihood of octave displacement for each voice.

Let's say we run the algorithm for 100 time steps, and then re-run it from the same initial values again for 100 time steps: due to the randomness in step 1 we will get different sequences (but with similar statistics). There is an important difference between sequences produced in this way and those produced by an 'un-feedback' random number generator with statistically independent (iid) trials. The statistical feedback reduces the sequence-to-sequence *fluctuations* (i.e. variance about the expectation) in the total number of occurrences of any given element over the 100 time steps, when compared to that of the un-feedback case. In Section 2.3 we will also show that, depending on the increment rule, the generated sequences exhibit different kinds of quasi-periodicity and "memory," as measured by decay of so-called *autocorrelation*, over durations much longer than the time to cycle through the n elements.

2.2 Formal description

By constructing a slightly more general model than sketched above, we open up an interesting parameter space varying from random to deterministic processes, that includes some of Tenney's work as special cases. For the remainder of Section 2 we assume more mathematical background.

For each element $i = 1, \dots, n$ we maintain a *count* c_i describing the number of time steps since that element was chosen. We define a single *growth function* $f : \mathbb{Z}^+ \rightarrow \mathbb{R}^+$ which acts on these counts to update the relative selection probabilities following each trial. Usually we have $f(0) = 0$ (which forbids repeated elements), and f non-decreasing. We also *weight* each element $i = 1, \dots, n$ with a fixed positive number w_i whose effect is to bias the selection probabilities towards certain elements and away from others. We now present a pseudocode which outputs a list $\{a_t\}_{t=1}^T$ containing the element chosen at each timestep $t = 1, \dots, T$.

Dissonant Counterpoint Algorithm

input parameters: weight vector $\{w_i\}_{i=1}^n$, growth function f

initialize: $c_i = 1, i = 1, \dots, n$

for timestep $t = 1, \dots, T$ **do**

compute probabilities: $p_i = \frac{w_i f(c_i)}{\sum_{k=1}^n w_k f(c_k)}, i = 1, \dots, n$

randomly choose j from the set $1, \dots, n$ with probabilities $\{p_i\}_{i=1}^n$

update counts: $c_j = 0$, and $c_i = c_i + 1, i = 1, \dots, n, i \neq j$.

store chosen note in the output list: $a_t = j$

end for

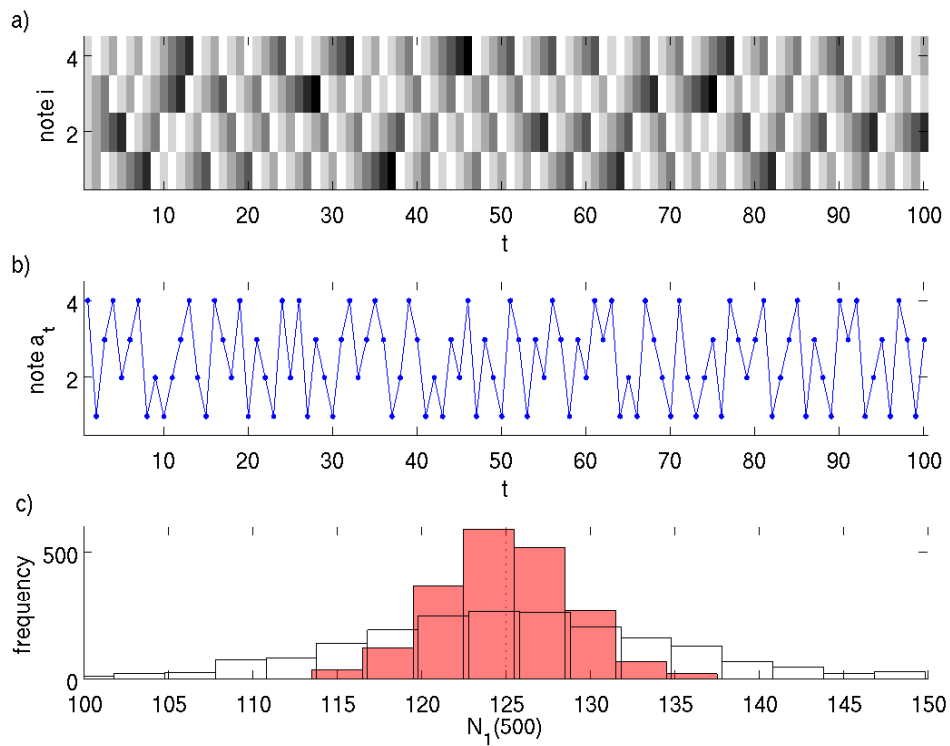


Figure 3: Simulation of the dissonant counterpoint algorithm for the simplest linear case (power $\alpha = 1$), and uniform weights $w_i = 1$ for $i = 1, \dots, 4$. a) grayscale image showing counts c_i for each element $i = 1, \dots, 4$ versus time t horizontally, with white indicating zero and darker larger counts. b) graph of elements selected a_t versus time t . c) distribution of $N_1(500)$, the total number of occurrences of element 1 in a run of 500 time steps, shown as a histogram over many such runs; shaded bars are for the dissonant counterpoint algorithm, white bars are for random iid element sequences with uniform distribution $p_i = 1/4$ for $i = 1, \dots, 4$ (note the much wider histogram).

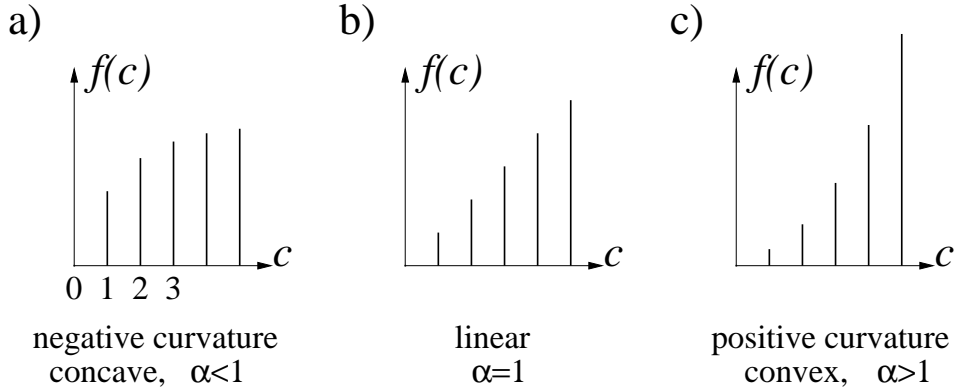


Figure 4: Illustration of three types of growth function f , depending on choice of power law α in (1).

The normalizing sum in the denominator of the expression for p_i merely ensures that the total probability is 1.

One simple but flexible form for the growth function is a *power law*,

$$f(c) = c^\alpha \quad (1)$$

for some power $\alpha \geq 0$. For now, we also fix equal weights $w_i = 1$ for all i . Note that $\alpha = 1$ gives the linear case where the relative selection probabilities are the counts themselves. A typical evolution of the counts c_i that form the core of the algorithm, for this linear case, is shown in Fig. 3a. A typical output sequence a_t is shown graphically in Fig. 3b (also see melody Fig. 1). A large reduction in variance of element occurrence statistics, relative to uniform random iid element sequences, is apparent in Fig. 3c.

Large powers, such as $\alpha > 5$, strongly favor choosing the notes with the largest counts, i.e. those which have not been selected for the longest time. Conversely, taking the limit $\alpha \rightarrow 0$ from above gives a process which chooses equally among the $n - 1$ notes other than the one just selected; because of its relative simplicity this version allows rigorous mathematical analysis (Section 2.4). Observe that $\alpha < 1$ leads to *concave* functions, i.e. with everywhere negative curvature (extending f to a function on the reals we would have $f'' < 0$), and that $\alpha > 1$ gives *convex* functions, positive curvature ($f'' > 0$). These cases are illustrated by Fig. 4, and their output is compared in the next section. In Fig. 5 we give a more complex musical example in which the power α , and hence the sonic and rhythmic texture, changes slowly during the piece.

Demonstration 2 (hocket for four percussionists)

Fast ♩ = 140-200 polansky

For four percussionists, five sounds each. Pick a logical distribution of sounds, for example:

lp 10/2/09
rev. 10/2/09

- one player wood (graduated in pitch), one player metal (similarly), one player skins, etc.
- the entire ensemble (20 distinct sounds) graduated in pitch, or loudness
- each player having one of each kind of sound, with some logical pitch (or unpitched) distribution
- completely arbitrary sound distribution

Any dynamics, but generally fairly static.

Figure 5: In this short quartet, 20 elements (percussion sounds) are selected by the algorithm, and distributed in hocket fashion to the four instruments. Durations are selected by the algorithm independently from a set of 9 distinct values. Durations and elements are selected independently, by different power functions of the form (1). The power α is interpolated over the course of the piece from 1 to some very high power (or vice versa). In the case of durations, the exponent begins high (maximum correlation) and ends at 1 (little correlation). In the case of the percussion elements, the interpolation goes in the other direction. An exponentially decreasing weight function is used for durations, favoring smaller values.

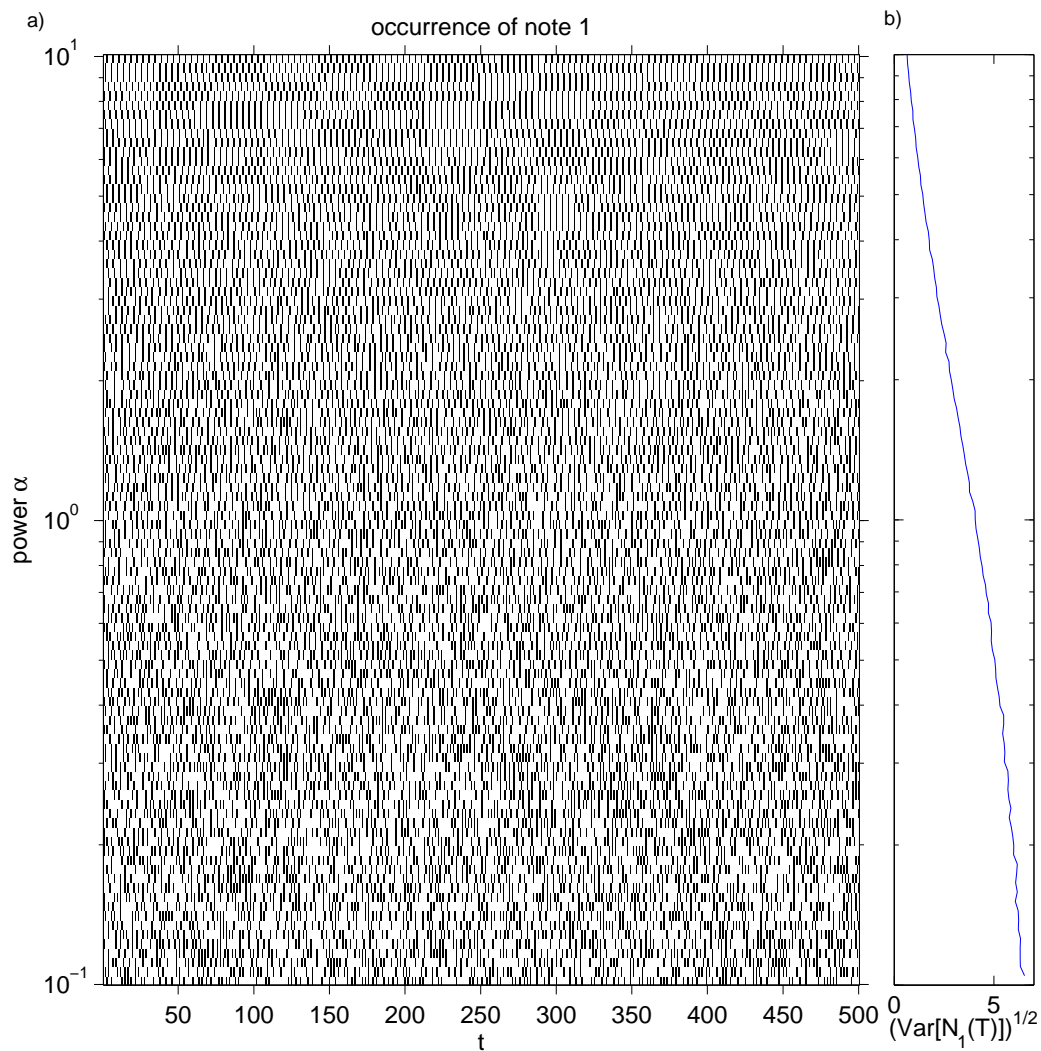


Figure 6: Dependence of periodic correlations on power law α . a) Occurrence of element 1 (black) for $n = 4$ elements, versus time $1 \leq t \leq T = 500$ horizontally, for 100 values of α spanning logarithmically the vertical direction from low (concave) to high (convex). Results are similar for the other elements. b) Standard deviation of the number of occurrences of element 1 in the run of length $T = 500$ (shown on a linear scale), as a function of α on the same vertical axis as in a).

2.3 Curvature of the growth function

In this section we investigate the effect of varying α , and introduce the autocorrelation function. We will take the weights w_i to be all equal. Figure 6a shows, for the case $n = 4$, sequences of algorithm outputs for a logarithmic range of α values. (Behaviors for other numbers of elements are qualitatively very similar.) The linear case, where $\alpha = 1$, lies half-way up the figure. Note that the larger α becomes, and consequently the more positive the curvature of the growth function, the more temporal order (repetitive quasi-periodic structure) there is in the occurrence of a given element. The sequences become highly predictable, locking into one of the $n!$ possible permutations of the n elements for a long period of time, then moving to another closely related permutation for another long period, and so forth. We may estimate⁹ this typical locked time period, when it, and n and α , are all large, as $e^{\alpha/n}$. Thus, to achieve the same level of temporal order with more elements, α would need to be increased in proportion to n .

What is the effect of varying α on the resultant statistics of a particular element? The number of times the element i occurs in a time interval (run) of length T is defined as

$$N_i(T) := \#\{t : a_t = i, 1 \leq t \leq T\} \quad (2)$$

$N_i(T)$ varies for each run of the algorithm, and is therefore a *random variable*, with mean (in the case of equal weights) T/n and variance $\text{Var}[N_i(T)]$. In Fig. 6b we show its standard deviation $(\text{Var}[N_1(T)])^{1/2}$ (indicating fluctuation size) against α , again for $n = 4$ and $T = 500$. This was measured using many thousand runs of this length. The standard deviation is large for low powers (concave case), and decreases by roughly a factor of 10 for $\alpha = 10$ (convex case). By comparison, the standard deviation for an iid uniform random sequence is larger than any of these: since $N_i(T)$ then has a binomial distribution with $p = 1/n$, it has standard deviation $\sqrt{Tp(1-p)} = 9.68 \dots$.

The change of “rhythmic patterning” seen above can be quantified using the *autocorrelation* in time. Consider the autocorrelation of the element signal a_t , defined as,

$$C_{aa}(\tau) := \lim_{T \rightarrow \infty} \frac{1}{T} \frac{\sum_{t=1}^T (a_t - \bar{a})(a_{t+\tau} - \bar{a})}{\text{Var}[a]} \quad (3)$$

where $\bar{a} = (n + 1)/2$ is the mean, and $\text{Var}[a]$ is the variance of the element signal. Fig. 7 shows $C_{aa}(\tau)$ plotted for four types of growth function, now

⁹This is done by estimating the ratio between the selection probabilities of the elements with $n - 1$ vs $n - 2$ counts as $[(n - 1)/(n - 2)]^\alpha \approx (1 + 1/n)^\alpha \approx e^{\alpha/n}$.

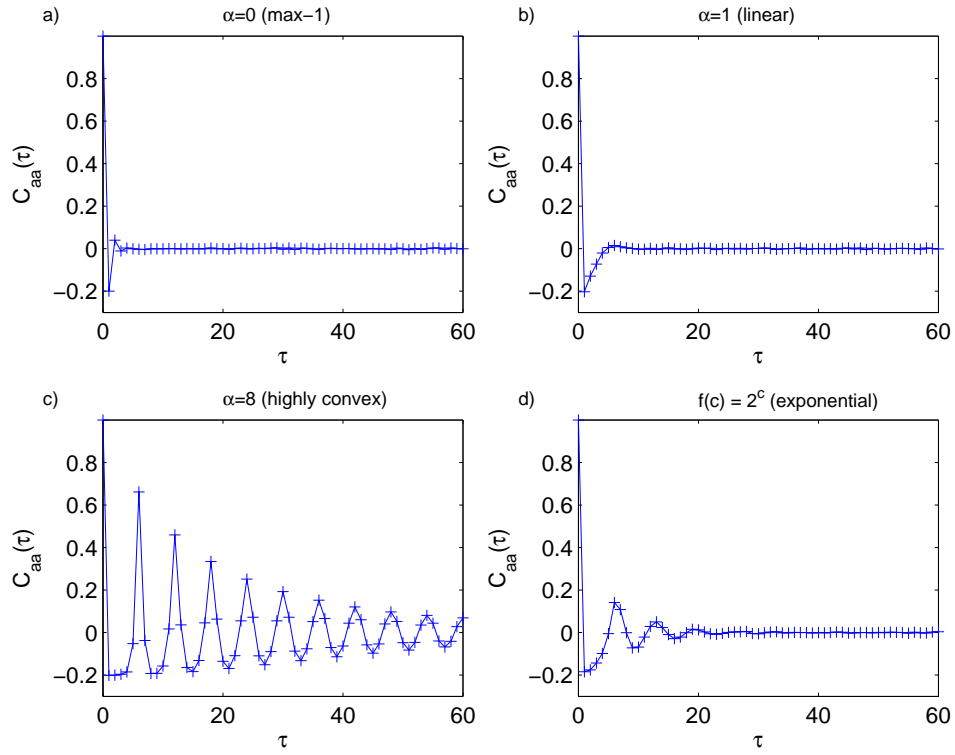


Figure 7: Autocorrelation functions measured from the dissonant counterpoint algorithm (with $n = 6$ elements, and equal weights) for four choices of growth function: three power laws, and an exponential law used in Section 3.1.

for $n = 6$. For $\alpha = 0$ we obtain the *max-1 rule* discussed in the next section: it has almost instantaneous decay of correlations, i.e. its “memory” is very short. For $\alpha = 1$, the linear model shows anti-correlation for the first few time steps, and virtually no memory beyond $\tau = n$. For high α (convex growth function), the tail of the autocorrelation extends much further (longer memory) up to $\tau = 50$ and beyond. Periodic peaks which soften with increasing τ also indicate quasi-periodic structure with a dominant period n . For this n , the exponential function (9) discussed in Section 3.1 gives correlation decay roughly similar to that of a power law model with $\alpha \approx 4$.

On a technical note, a well-known mathematical result from stochastic processes (related to Einstein’s fluctuation-dissipation relation in physics [11, Eq. (2.1)]) equates the rate of growth with T of $\text{Var}[N_i(T)]$ to the area under (or sum over τ of) the autocorrelation function graph.¹⁰ It is surprising that the above results on variance of $N_1(T)$ imply that for large α the signed area under the autocorrelation graph is actually *smaller* than for small α , despite the fact that the tails extend to much *longer* times τ . It would be interesting to find an explanation for this.

2.4 Vanishing power: “max-1” version and the effect of weights

When $\alpha = 0$, the growth function (1) becomes the function

$$f(c) = \begin{cases} 0, & c = 0 \\ 1, & c = 1, 2, \dots \end{cases} \quad (4)$$

This *max-1 rule* is the linear algorithm truncated at a maximum value of 1. As discussed above, its element statistics have a large variance and an almost instant decay in the autocorrelation (i.e. almost no memory). The algorithm chooses between all $n - 1$ notes other than the current one, weighted only by their corresponding weights w_i . For convenience, in this section we assume these weights have been normalized, thus

$$\sum_{i=1}^n w_i = 1 \quad (5)$$

Since no account is taken of the number of *counts* each eligible note has accumulated, the algorithm becomes what is known as a *Markov chain* [28],

¹⁰Strictly speaking, the relevant area is $\sum_{\tau \in \mathbb{Z}} C_{ii}(\tau)$ where C_{ii} is the autocorrelation of the binary signal which is 1 when element i is selected, and 0 when any other element is chosen. We have checked that $C_{ii}(\tau)$ and $C_{aa}(\tau)$ look very similar.

with no explicit memory of anything other than the current selected element. Its n -by- n transition matrix M then has nonnegative elements

$$M_{ij} = \begin{cases} \frac{w_i}{1-w_j}, & i \neq j \\ 0, & i = j \end{cases} \quad (6)$$

which give the probability of element i being selected given that the current element is j . By using (5) one may verify the required column sum rule $\sum_i M_{ij} = 1, \forall j$.

Recall that weights w_i were included in the algorithm to give long-term bias towards various elements. So, how do the long-term frequencies of elements depend on these weights? The relationship is not trivial: frequencies are not strictly proportional to weights. The relative frequencies tend to the Markov chain's so-called *steady-state* probability distribution $\mathbf{p} := \{p_i\}_{i=1}^n$ (whose components p_i are normalized $\sum_{i=1}^n p_i = 1$), for which we can solve¹¹ as follows.

Theorem 1 *The max-1 rule with normalized weights $\{w_i\}_{i=1}^n$ and Markov transition matrix (6) has a unique steady state distribution given by*

$$p_i = \frac{w_i(1-w_i)}{\sum_{j=1}^n w_j(1-w_j)}, \quad i = 1, \dots, n \quad (7)$$

Proof: Consider a candidate distribution vector $\mathbf{v} \in \mathbb{R}^n \setminus \mathbf{0}$. We multiply \mathbf{v} by a non-zero scalar to give it the more convenient weighted normalization $\sum_{i=1}^n v_i/(1-w_i) = 1$. Using this and (6) we compute the i th component of $(M - I)\mathbf{v}$ as follows,

$$\begin{aligned} (M\mathbf{v} - \mathbf{v})_i &= w_i \sum_{j \neq i} \frac{v_j}{1-w_j} - v_i = w_i \left(1 - \frac{v_i}{1-w_i} \right) - v_i \\ &= w_i - \frac{1}{1-w_i} v_i \end{aligned}$$

The condition that this vanish for all $i = 1, \dots, n$, in other words $v_i = w_i(1-w_i)$, is equivalent to the statement that \mathbf{v} is an eigenvector of M with eigenvalue 1 and therefore an (unnormalized) steady state vector. Hence the eigenvector is unique up to a scalar multiple, i.e. this eigenvalue is simple.

¹¹Some intuition as to why an analytic solution is possible here is that M may be factored as the product of three simple matrices: $M = \text{diag}\{w_i\}(1 - I)\text{diag}\{(1 - w_i)^{-1}\}$, where $\mathbf{1}$ is the n -by- n matrix with all entries 1.

Finally, normalizing (in the conventional sense) this formula for v_i gives the expression (7). \square

In other words, with weighted versions of the algorithm, it turns out that statistical differences in element frequencies are less pronounced than the corresponding differences in weights.

How frequently may one of the n elements occur? Even if we push one of the weights towards 1 at the expense of the others (which must then approach zero), the corresponding element may occur no more than $1/2$ of the time. Intuitively, this follows since repeated elements are forbidden, so one element can be chosen at most every other timestep. Rigorously, we have the following.

Theorem 2 *Let $n \geq 3$. Then the max-1 rule with positive weights $\{w_i\}_{i=1}^n$ has a steady state distribution whose components obey $p_i < 1/2$ for $i = 1, \dots, n$.*

Proof: Fixing i , we have, using the fact that $1 - w_j > w_i$ for all $j \neq i$,

$$\begin{aligned} \sum_{j=1}^n w_j(1 - w_j) &= w_i(1 - w_i) + \sum_{j \neq i} w_j(1 - w_j) \\ &> w_i(1 - w_i) + w_i \sum_{j \neq i} w_j \end{aligned}$$

The result then follows from $\sum_{j \neq i} w_j = 1 - w_i$ and Theorem 1. \square

For the case $n = 2$, the possibility $p_i = 1/2$ must also be allowed. The result carries over to general growth functions f with $f(0) = 0$, again for the simple reason that repetition is excluded. If, however, the “drop-down” value $f(0)$ (the value to which a selected element is reset prior to the next selection) is greater than zero, repetition becomes possible. In some of Tenney’s music (such as the piece about which he first published a description of this algorithm, *Changes*), he specifies a “very small” drop-down value (see Section 3).

Finally, we note that a wide variety of complex behaviors can result from combining the convex (large- α) power law with unequal weights. This seems to result in a competition between the tendency for locked-in permutations of all n elements due to the large α , and the strong bias for heavily-weighted elements. For example, Fig. 8 illustrates a selection process among $n = 12$ elements, with weights strongly biased towards “high” elements. The

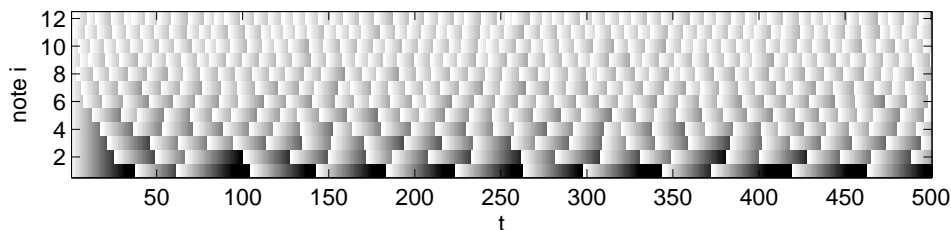


Figure 8: Grayscale image of counting function produced for the power-law with $\alpha = 6$ and $n = 12$ pitch-classes, with highly unequal weights $w_i = i^5$. $T = 500$ time steps are shown. This is discussed in Sec. 2.4.

resulting behavior consists of disordered clusters of arpeggiated sequences. It is striking that even with such extremely unequal weights (element 12 is $12^5 = 248832$ times more preferred than element 1), element 12 only occurs a few times more often than element 1.

3 Examples from Tenney’s Work

Tenney’s interest in the ideas of dissonant counterpoint dates back to the 1950s, as evidenced by pieces like *Seeds* (for ensemble) (1956; revised 1961) and *Monody* (for solo clarinet) (1959) [18]. These pieces, while through-composed without the use of a computer, show his nascent fascination with achieving what he later refers to—with respect to the early electronic works—as “variety”:

If I had to name a single attribute of music that has been more essential to my esthetic than any other, it would be *variety*. It was to achieve greater variety that I began to use random selection procedures in the Noise Study (more than from any philosophical interest in indeterminacy for its own sake), and the very frequent use of random number generation in all my composing programs has been to this same end. [30] (p. 40)

Tenney began using the computer for his compositions in 1961. These works, produced at Bell Laboratories, are among the first examples of computer music. Most dealt primarily with both the new possibilities of computer synthesis and his ideas of hierarchical temporal gestalt formation [38, 29]. Yet he recognized that randomly generated events without memory

of prior events would not produce the variety (Cowell: non-tautology) that he wanted.

While the early computer pieces predate a formalization of the dissonant counterpoint algorithm, the “seeds” of this idea are clear in his description of an approach to pitch selection:

Another problem arose with this [Stochastic String] quartet which has led to changes in my thinking and my ways of working, and may be of interest here. Since my earliest instrumental music (“Seeds,” in 1956), I have tended to avoid repetitions of the same pitch or any of its octaves before most of the other pitches in the scale of 12 had been sounded. This practice derives not only from Schoenberg and Webern, and 12-tone or later serial methods, but may be seen in much of the important music of the century (Varèse, Ruggles, etc).

In the programs for both the Quartet and the Dialogue, steps were taken to avoid such pitch-repetitions, even though this took time, and was not always effective (involving a process of recalculation with a new random number, when such a repetition did occur, and this process could not continue indefinitely). In the quartet, a certain amount of editing was done, during transcription, to satisfy this objective when the computer had failed. [30]

Tenney continued to explore this process throughout his life, and began using the algorithm described in this paper as early as the 1980s. The first published description of it occurs in a sentence in his article on *Changes* (1985):

Just after a pitch is chosen for an element, [the probability of] that pitch is reduced to a very small value, and then increased step by step, with the generation of each succeeding element (at any other pitch), until it is again equal to 1.0. The result of this procedure is that the immediate recurrence of a given pitch is made highly unlikely (although not impossible). [34] (p. 82).

Note that the above description seems to describe a linear model with a small positive “drop-down” value, and truncation, i.e.

$$f(c) = \min[\epsilon + ac, 1] \tag{8}$$

for $a > 0$ and some small $\epsilon > 0$. The fact that ϵ is positive allows pitches to be repeated.

From the composition of *Changes* in 1985 until 1995, Tenney wrote a number of computer-generated pieces, including *Road to Ubud* (1986; revised 2001), *Rune* (1988), *Pika-Don* (“flash-boom”) (1991) and *Stream* (1991), each of which warrant further investigation with respect to the use of the dissonant counterpoint algorithm. However, many of Tenney’s works after 1995 implement the dissonant counterpoint algorithm explicitly including: *Spectrum 1 – 8* (1995 – 2001); *Diaphonic Study* (1997); *Diaphonic Toccata* (1997); *Diaphonic Trio* (1997); *Seegersong #1* and *#2* (1999); *Prelude and Toccata* (2001); *To Weave (a meditation)* (2003); *Panacousticon* (2005); and *Arbor Vitae* (2006).¹²

At a certain point, the dissonant counterpoint algorithm simply became Tenney’s de facto pseudo-random element chooser. He used it to determine pitches (*Seegersongs* and others), timbre/instrumentation (*Spectrum* pieces, *Panacousticon*), and register (*To Weave*), and even movement through harmonic space (*Arbor Vitae*). Early drafts of computer programs written to generate *Spectrum 6 – 8* are labeled with the word *diaphonic*. Tenney used that term to refer to most of his computer code after about 1995. Specific titles notwithstanding, he may have considered many or all of these works as “diaphonic” studies after Ruth Crawford Seeger’s four studies from the early 1930s (taking their name from Charles Seeger’s earlier use of the term to mean, roughly: “sounding apart” [27]). Over time, the algorithm’s role seems to change from that of a principal formal determinant (as in the *Seegersongs* and *To Weave*) to an embedded, deep-level selection technique which was combined with and modulated by larger formal processes.

3.1 *Seegersongs*

Seegersong #1 and *#2* are perhaps the clearest examples of Tenney’s use of the algorithm. These pieces exemplify Tenney’s integration of the algorithm with larger formal concerns. Both *Seegersongs* used the convex growth function that Tenney most commonly employed in these later works, in this case by repeated doubling, thus

$$f(c) = 2^c \tag{9}$$

Seegersong #1 and *#2* explicitly model Ruth Crawford Seeger’s approach to dissonant counterpoint in the avoidance of pitch-class repetition. However, they also suggest other aspects of her work, such as the technique of “phrase structure” discussed by Charles Seeger in the “Manual...” [26]

¹²For these latter pieces (unlike those between 1985–1995), the computer code is available. Other pieces may use the algorithm in some way that is not yet known.

and exemplified by Ruth Crawford Seeger in *Piano Study in Mixed Accents* (1930/31) as well as by Johanna Beyer in her two solo clarinet suites (1932) [10, 9, 22]. As with Seeger’s *Piano Study in Mixed Accents*, both Tenney’s *Seegersongs* consist of delimited phrases (or, in Tenney’s terminology, *gestalt sequences*). Each phrase has an associated ascending or descending pitch-trajectory over some duration. This is achieved using a generalization of the dissonant counterpoint algorithm of Section 2.2, in which the weights w_i are allowed to change with time in a prescribed fashion, and therefore are labeled $w_{i,t}$. That is, the probabilities in the above algorithm are computed according to

$$p_i = \frac{w_{i,t}f(c_i)}{\sum_{k=1}^n w_{k,t}f(c_k)}, \quad i = 1, \dots, n \quad (10)$$

Tenney used weights $w_{i,t}$ which decrease linearly with pitch distance from a pitch center, giving a triangularly-shaped weight vector which reaches zero a pitch distance of a tritone from the center. The center itself moves in time in a piecewise linear fashion, with each linear trajectory being a phrase. The resulting weights are shown as grayscale density in Fig. 9. The moving weights are used by the dissonant counterpoint algorithm to follow the desired registral trajectory. The interpolation points defining the linear trajectories are themselves chosen randomly within a slowly-changing pitch range illustrated by the gray region in Fig. 10b.

The large scale form of Tenney’s *Seegersongs* resembles Ruth Crawford Seeger’s *Piano Study in Mixed Accents*, in which the registral profile similarly ascends and then descends (Fig. 10a). However, in Tenney’s re-imagining, the range’s upper limit follows the positive part of a smoothly distorted cosine function peaking at the golden mean division of the piece’s duration. The lower limit of the pitch range remains constant (see Fig. 10b).

3.2 The *Spectrum* pieces

The pitches in the *Spectrum* series are derived from a harmonic series with a fixed fundamental [15]. In these pieces, Tenney also used the algorithm to determine non-pitch parameters. In the *Spectrum* works that use percussion, the algorithm selects, for those instruments, from a set of pitched and unpitched sounds. When the algorithm selects a pitch that cannot be played accurately by a pitched percussion instrument, a number is returned indicating an unpitched percussion sound. In this case, “accurate” is defined as a pitch in equal-temperament that is more than 5 cents from its cognate harmonic. In the instructions, Tenney states that “numbers in place of note-heads, denote non-pitched sounds or instruments to be freely chosen by the

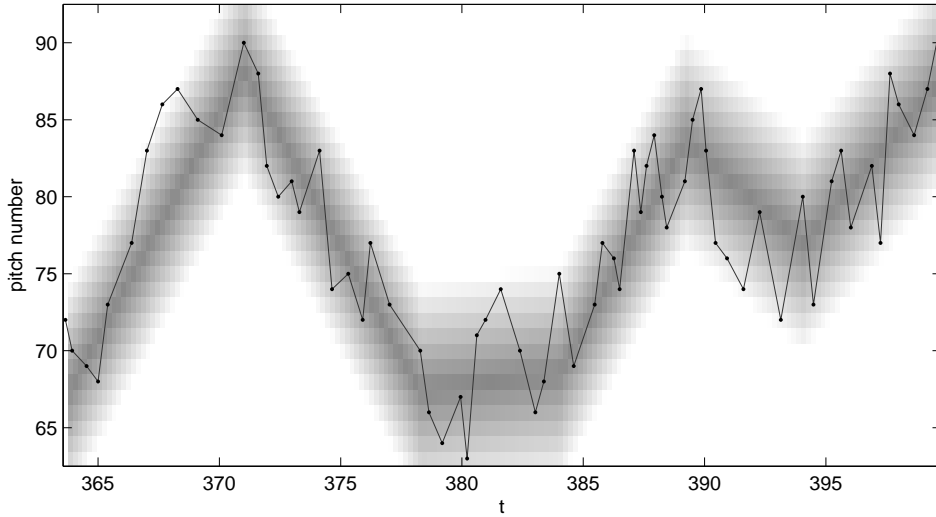


Figure 9: *Seegerson #2* excerpt showing pitch profile (solid lines) and dynamic weights $w_{i,t}$ (grey density: white is zero and darker larger positive values) used in the dissonant counterpoint algorithm (see text for how $w_{i,t}$ is generated).

player” [36].

In the *Spectrum* pieces with piano and/or harp, those fixed-pitch instruments are retuned and thus not subject to the process described above. However, because these instruments are polyphonic, the counts for more than one pitch (in the selection process) are reset to zero simultaneously. That is, all the notes for the chord are treated as having been selected. Tenney chooses the *number of chord tones* stochastically, based on a function of upper and lower density limits over time. All other parameters of the *Spectrum* pieces such as duration, loudness and pitch (which integrates the dissonant counterpoint algorithm) are determined in similar ways (for more on parametric profiles see [29]).

For the note selection algorithm, the *Spectrum* pieces use a growth function similar to that of the *Seegersons*, but with a larger base for exponential growth:

$$f(c) = 4^c \quad (11)$$

Each part in each *Spectrum* piece is generated individually. Since the larger the base, the more convex the function (this being similar to the effect of a larger power explored in Section 2.3), note selection will tend to be

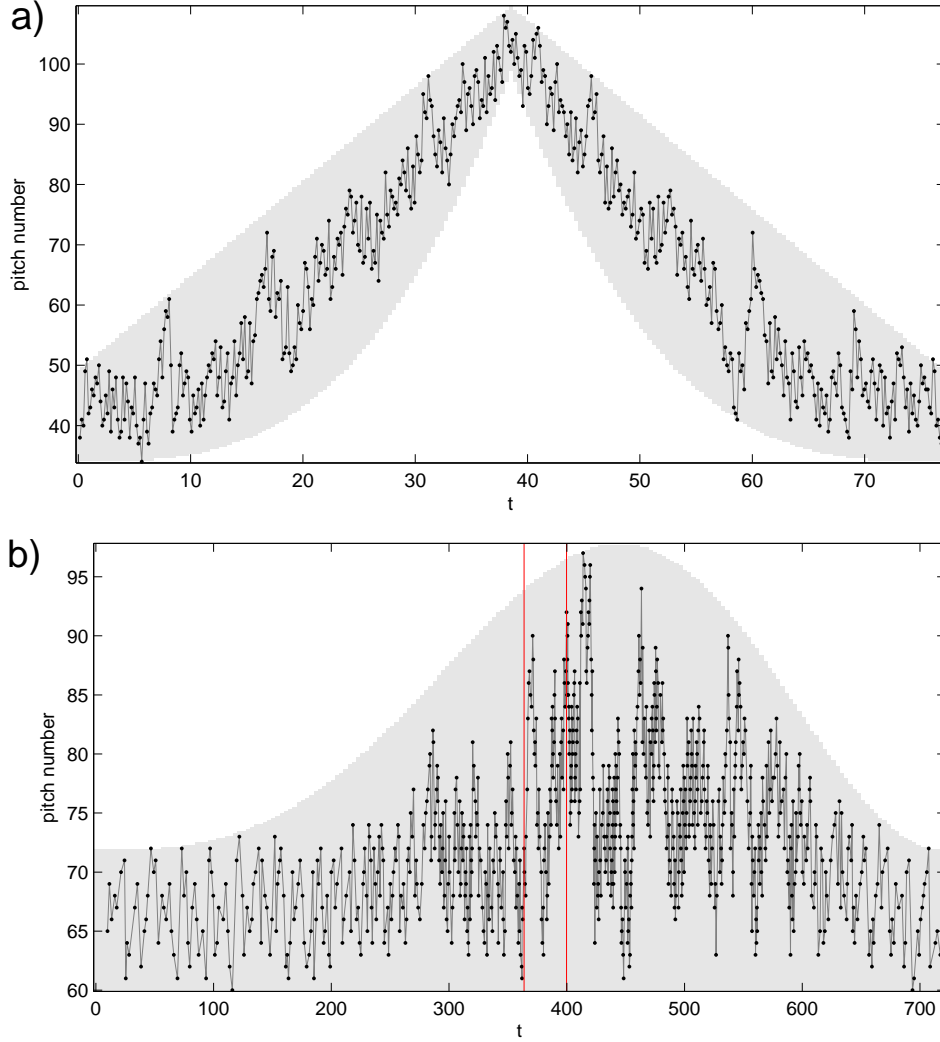


Figure 10: Pitch profiles of a) Ruth Crawford Seeger’s *Piano Study in Mixed Accents* (less than a minute and a half long, about a 6 octave range), compared against b) Tenney’s *Seegersong #2* (12 minutes long, about a 3 octave range). In each case time is horizontal (seconds) and pitch vertical (semitones, where 60 is middle C). The gray region shows the time-dependent pitch range used; see text for discussion of the algorithms. In b) the upper bound is proportional to $25 - 13 \cos[2\pi(t/t_{\max})^{1.44}]$, and the vertical lines show the start and end of the excerpt in Fig. 9.

more correlated than in the *Seegersongs*. Perhaps, because there is a large variety of instruments in the *Spectrum* pieces, Tenney might have used the higher base in order to give individual voices a more coherent, even melodic character.

3.3 *To Weave (a meditation)*

In *To Weave (a meditation)*, for solo piano, the selection algorithm determines not only pitch-class but also register, or what can be called “voice.” Three such voices are used in the piano part, with the max-1 algorithm of Section 2.4.

The algorithm determines voices note by note, where a voice is defined as one of three registers (low, middle or high). Each voice is thus a possible element for selection. For each note, the two voices not selected for the previous note become equally probable and the selected voice’s probability is set to 0. In other words, if a pitch occurs in the low register, then the next pitch must occur within one of the two other registers (middle or high). The stochastic pitch sequence is *woven* “non-tautologically” into a three-voice virtual polyphony (thus the titular pun on the name of the pianist, Eve Egoyan, for whom it was written).

The growth function (2^c) is the same as the *Seegersongs*, but in *To Weave* pitch-class probabilities are incremented both globally and locally (for each individual voice). These two values are multiplied to determine the pitch-class probabilities ultimately used to select a pitch-class. Once selected, that pitch-class is placed within the range of the currently selected voice. As in the *Seegersongs*, the ranges of the three voices change over time, peaking at the golden mean point of the piece. According to Tenney:

Waves for Eve, wave upon wave, little waves on bigger waves, et cetera, but precisely calibrated to peak at the phi-point of the golden ratio. To weave: a three-voice polyphonic texture in dissonant counterpoint, with a respectful nod in the direction of Carl Ruggles and Ruth Crawford Seeger. [37]

3.4 *Panacousticon*

In *Panacousticon* for orchestra, the algorithm selects both pitch-class and instrument. As in the *Spectrum* pieces, the pitches are derived from the harmonic series on one fundamental. Both implementations of the algorithm



Figure 11: *To Weave (a meditation)* score excerpt.

(for pitch-class and instrumentation) use linear growth functions with an upper bound, of the form

$$f(c) = \min[c, 5] \quad (12)$$

Thus after an element is chosen, its probability reaches a maximum if it is not chosen within the next five selections.¹³ For each note, the dissonant counterpoint algorithm is combined with another procedure that determines the register of the chosen pitch-class and which instruments are available to play the pitches (that is, the instruments that are not already sounding and whose range covers the determined pitch).

3.5 *Arbor Vitae*

In his last work, *Arbor Vitae* for string quartet, Tenney uses the algorithm to explore complex harmonic spaces using “harmonics of harmonics,” in perhaps his most complex and unusual usage of this selection method. The pitch material, or the *harmonic space* (see, for example [33]) of *Arbor Vitae*

¹³Generally, the effect of such an upper bound on f is to equalize the probabilities of all elements not chosen within the upper bound number of time steps. This may be viewed as negative curvature as in Fig. 4, and it serves to reduce the already small autocorrelation of the resulting sequence at large times. An upper bound of 1 would be the same as a power law $\alpha = 0$, and is the max-1 rule of Section 2.4.

is complex. However, ratios are generated via reasonably simple procedures, in a manner which recalls Lou Harrison’s idea of “free style” just intonation (such as in *A Phrase from Arion’s Leap*, *Symphony in Free Style*, and *At the Tomb of Charles Ives*; see [16, 23, 19, 20, 17]).

The title is a metaphor for the work’s harmonic structure [40]. The dissonant counterpoint algorithm first calculates *roots*, which are harmonics of a low B-flat. These are treated as temporary, *phantom* fundamentals. Next, the algorithm calculates *branches* which terminate in the pitch-classes for sounding tones. Lower harmonics are biased for root calculation by assigning *initial* probabilities (p_i proportional to $\frac{1}{\sqrt{i}}$ where i is the harmonic number of the possible root).¹⁴ The growth function depends on harmonic number i . The selection algorithm can be summarized as that of Section 2.2 but with the probabilities computed via

$$p_i = \frac{f_i(c_i)}{\sum_{k=1}^n f_k(c_k)} \quad (13)$$

where $f_i(c)$ now depends both on element i and counts c , as follows,

$$f_i(c) = \begin{cases} 0, & c = 0 \\ i^{-1/2}, & c = 1 \\ i^{-1/4}, & c > 1 \end{cases} \quad (14)$$

Initial counts c_i are all set to 1, and counts are only updated for elements that have been selected at least once.

For branch selection, the set of possible elements are the primes — 3, 5, 7, 11 — of a given root. Due to the use of negative powers, the growth function becomes a kind of harmonic distance measure (with higher primes less favored). This tendency towards “consonance” is reflected in the bias towards selected roots which are closer to fundamentals, as well as in selected branches which are closer to those roots. Tenney seems to be imposing a kind of natural evolutionary “drag” on the tendency of the harmonic material in the piece to become too strongly disassociated with its fundamentals. This ensures a kind of tonality, albeit a sophisticated and ever-changing one.

In other ways, compositional procedures of *Arbor Vitae* resemble those of *Seegersongs*, *Panacousticon*, *To Weave (a meditation)*, and the *Spectrum* pieces. Final pitch determinations use time-variant profiles of pitch ranges. Instrument selection, as in some of the other pieces, is performed by the dissonant counterpoint algorithm. As in *Panacousticon*, the software first determines whether an instrument’s range can accommodate the selected pitch.

¹⁴In fact, rather than handling counts as in Section 2.2, Tenney *directly* updated relative probabilities p_i , normalizing them to sum to 1 whenever random selection was needed.

In contrast to *To Weave (a meditation)*, the growth function for instrument selection in *Arbor Vitae* has $f(0) = f(1) = 0$: in general no instrument may be chosen until *two* other instruments have played. For a more detailed discussion of this piece, see [5].

4 Conclusion

The dissonant counterpoint algorithm is, in some respects, just a simple method for choosing from a set of elements to give a random sequence with certain behaviors. In other respects, it is an ingenious way of marrying both an important historical style (that of Ruggles/Seeger/Cowell “dissonant counterpoint,” or “diaphony”) with a more modern and sophisticated, but poorly understood set of ideas from computer-aided music composition (Ames’ *statistical feedback*). The algorithm elegantly embeds the latter idea in a manner characteristic of Tenney’s interest in the idea of a *model* (a method that reflects how humans do or hear something).

We have analyzed the algorithm mathematically, explaining how a convex/concave choice of the growth function controls the correlation in time (rate of memory loss) of the sequence. This can lead to surprising musical effects, such as quasi-rhythmic permutations with long-range order. We included a formula explaining how weights determine the statistical selection frequencies (in the case of the max-1 growth function). We illustrated a few of the algorithm’s wide variety of musical possibilities with two example compositions. Combined with our discussion of the algorithm’s role in Tenney’s work, this work suggests ways in which it might be used further, in experimental and musical contexts.

For example, none of the versions of the algorithm presented above consider an “ordered” set of elements in which the *proximity* of one to the other is significant. Simple examples of this are pitch-sets, registral values, durations, and so on. One might want to select from meaningful *regions* of the set of elements (e.g. “shorter durations,” “higher pitches”). In a simple extension to the algorithm, one of the authors (Polansky) has implemented what he calls “gravity”: not only is the chosen element’s probability affected post-selection, but so are the probabilities of surrounding elements. By defining the shape of the “gravitation” (the width of the effect, the slope of the effect curves, whether the “gravity well” is negative or positive) one can increase or decrease the probabilities of neighborhood selection in a variety of ways. Polansky has used this extensively in his piece *22 Sounds* (for percussion quartet) [24].

The algorithm, in its simplest form, is meant to ensure a kind of maximal variety with a minimum amount of computation. However, by varying explicit parameters, it can produce, in both predictable and novel ways, a continuum of behaviors from completely non-deterministic to completely deterministic.

Acknowledgements

Thanks to Dan Rockmore for his important help in describing the relationship of the mathematics to the aesthetic ideas here. Kimo Johnson made many valuable suggestions, and helped work on some of the software. Thanks to Lauren Pratt and the Estate of James Tenney for making valuable material available to us. AHB is funded by NSF grant DMS-0811005.

A Matlab/Octave code examples

Here we give code that we use to simulate the dissonant counterpoint algorithm and collect statistics. It may be run with MATLAB [39] or its free alternative Octave [14]. To measure autocorrelation accurately we need many (e.g. 10^5) samples. We generate N realizations of length T simultaneously, since this is equivalent to, but (due to vectorization) much more efficient than, generating a single long realization of length NT . This also allows us to see ‘vertical’ correlations between different runs launched with the same initial conditions (as in Fig. 6).

```

n = 12;                % # notes or elements
N = 500;              % # simultaneous realizations
T = 500;              % # timesteps
f = @(c) c.^4;        % fourth power law growth function (convex)
w = ones(n,1);        % selection bias weights (column vector)
a = zeros(N,T);       % histories of which notes chosen
c = ones(n,N,T+1);    % histories of counts for all notes
for t=1:T
    fc = repmat(w, [1 N]) .* f(c(:,:,t));          % feed c thru func & bias
    p = fc ./ repmat(sum(fc,1), [n 1]);             % selection probabilities
    cum = cumsum(p,1);                               % cumulative probs
    x = rand(1,N);                                    % random iid in [0,1]
    a(:,t) = sum(repmat(x, [n 1]) > cum, 1) + 1;    % chosen notes
    c(:,:,t+1) = c(:,:,t) + 1;                       % increment counts
end

```

```

    c(sub2ind(size(c), a(:,t)', 1:N, (t+1)*ones(1,N))) = 0; % reset chosen
end
figure; imagesc(a); colorbar; xlabel('t'); ylabel('run'); title('notes');

```

To compute autocorrelation of the element sequences a we use,

```

M = 100; % max correlation time to explore
ma = mean(a(:)); zma = a - ma; ca = zeros(1,M+1); l = 1:(T-M);
for t=0:M, ca(t+1) = mean(mean(zma(:,l).*zma(:,l+t))); end
figure; plot(0:M, ca./ca(1), '+-'); xlabel('\tau'), ylabel('C_a(\tau)');

```

Finally, to generate an audio file output of realization number r we use,

```

dt = 0.125; fs = 44100; Tsong = 60; % Tsong: song length (sec)
fnot = 440*2.^((0:n-1)/n); % list of note freqs (Hz)
t = single(1:floor(fs*Tsong)-1)/fs; % list of time ordinates
wavwrite(0.9*sin(2*pi*fnot(a(r,1+floor(t/dt))).*t)', fs, 16, 'out.wav');

```

References

- [1] Charles Ames, *Two pieces for amplified guitar*, Interface: Journal of New Music Research **15** (1986), no. 1, 35–58.
- [2] ———, *Automated composition in retrospect*, Leonardo Music Journal **20** (1987), no. 2, 169–185.
- [3] ———, *Tutorial on automated composition*, Proceedings of the ICMC, International Computer Music Association. Urbana, Illinois, 1987, pp. 1–8.
- [4] ———, *A catalog of statistical distributions: Techniques for transforming random, determinate and chaotic sequences*, Leonardo Music Journal **2** (1991), 55–70.
- [5] ———, *Statistics and compositional balance*, Perspectives of New Music **28** (1991), no. 1, 80–111.
- [6] ———, *A catalog of sequence generators*, Leonardo Music Journal **2** (1992), 55–72.
- [7] ———, *Thresholds of confidence: An analysis of statistical methods for composition, part 1: Theory*, Leonardo Music Journal **5** (1995), 33–38.

- [8] ———, *Thresholds of confidence: An analysis of statistical methods for composition, part 2: Applications*, Leonardo Music Journal **6** (1996), 21–26.
- [9] Marguerite Boland, *Experimentation and process in the music of Johanna Beyer*, VivaVoce, Journal of the Internationaler Arbeitskreis Frau und Musik (in German) **76** (2007), <http://www.archiv-frau-musik.de>.
- [10] ———, *Johanna Beyer: Suite for clarinet Ib*, Frog Peak Music (A Composers' Collective), Hanover, NH, 2007, Annotated performance edition.
- [11] L. A. Bunimovich, *Existence of transport coefficients*, Encyclopaedia Math. Sci., vol. 101, pp. 145–178, Springer, 2000.
- [12] Michael Casey, *HS: A symbolic programming language for computer assisted composition*, Master's thesis, Dartmouth College, 1992.
- [13] Henry Cowell, *New musical resources*, Something Else Press Edition (reprinted 1969), 1930.
- [14] John W. Eaton, *GNU Octave manual*, Network Theory Limited, 2002, <http://www.octave.org>.
- [15] Robert Gilmore, Liner notes to *James Tenney: Spectrum Pieces*, CD, New World Records 0692, 2009.
- [16] Lou Harrison, *A Phrase for Arion's Leap*, Recording on *Tellus #14: Just Intonation*, (re-issued recording, 1986), 1974.
- [17] Leta Miller and Fred Lieberman, *Lou Harrison: Composing a world*, Oxford University Press, 1998.
- [18] Larry Polansky, *The Early Works of James Tenney*, Soundings (Peter Garland, ed.), vol. 13, Santa Fe: Soundings Press, 1983.
- [19] ———, *Item: Lou Harrison as a speculative theorist*, A Lou Harrison Reader (Peter Garland, ed.), Soundings Press, 1987.
- [20] ———, *Paratactical Tuning: An agenda for the future use of computers in experimental intonation*, Computer Music Journal **11** (1987), no. 1, 61–68.

- [21] ———, *More on morphological mutations*, Proceedings of the ICMC, Interational Computer Music Association, San Jose, 1992, pp. 57–60.
- [22] ———, Liner notes to “*Sticky Melodies*”: *The Choral and Chamber Music of Johanna Magdalena Beyer*, CD, New World Records 80678, 2008.
- [23] ———, *Three Excellent Ideas*, 2009, Talk given at the Drums Along the Pacific Festival, Cornish Institute for the Arts, Seattle, http://eamusic.dartmouth.edu/~larry/misc_writings/talks/three_excellent_ideas/three_excellent_ideas.front.html.
- [24] ———, *22 Sounds (for percussion quartet)*, 2010.
- [25] ———, *No Replacement (85 Verses for Kenneth Gaburo)*, Perspectives of New Music **33** (Winter–Summer, 1995), no. 1–2, 78–97.
- [26] Charles Seeger, *Manual of dissonant counterpoint*, Studies in Musicology II: 1929-1979 (Ann Pescatello, ed.), Berkeley: University of California Press (republished 1994), 1930.
- [27] ———, *On Dissonant Counterpoint*, Modern Music **7** (June–July, 1930), no. 4, 25–31.
- [28] D. Stroock, *An introduction to Markov processes*, Springer, 2005.
- [29] James Tenney, *Meta + Hodos*, Oakland, California: Frog Peak Music, 1964, (republished 1986).
- [30] ———, *Computer Music Experiences, 1961–1964*, Electronic Music Reports **1** (1969).
- [31] ———, *The chronological development of Carl Ruggles’ melodic style*, Perspectives of New Music **16** (1977), no. 1, 36–69.
- [32] ———, *Conlon Nancarrow’s Studies for player piano*, Conlon Nancarrow: Selected Studies for Player Piano, Berkeley, California: Soundings Press, 1977.
- [33] ———, *John Cage and the Theory of Harmony*, Soundings (Peter Garland, ed.), vol. 13, Santa Fe: Soundings Press, 1983.
- [34] ———, *About ‘Changes’: Sixty-four studies for six harps*, Perspectives of New Music **25** (1987), no. 1–2, 64–87.

- [35] ———, Liner notes to *James Tenney: Bridge and Flocking*, CD, hat ART CD 6193, 1996.
- [36] ———, *Spectrum 8 (for solo viola and six instruments)*, Hanover, NH: Frog Peak Music (A Composers' Collective), 2001.
- [37] ———, Liner notes to *Weave: Eve Egoyan*, CD, EVE0106, 2005.
- [38] James Tenney and Larry Polansky, *Hierarchical temporal gestalt perception in music: a metric space model*, *Journal of Music Theory* **24** (1980), no. 2, 205–241.
- [39] The MathWorks, Inc., *MATLAB software*, Copyright (c) 1984–2009, <http://www.mathworks.com/matlab>.
- [40] Michael Winter, *On James Tenney's 'Arbor Vitae' for string quartet*, *Contemporary Music Review* **27** (2008), no. 1, 131–150.

Selected Discography of Cited Tenney Compositions

- [41] Quatuor Bozzini, *Arbor Vitae*, CD, CQB 0806, 2008.
- [42] Eve Egoyan, *Weave: Eve Egoyan*, CD, EVE0106, 2005.
- [43] Maelstrom Percussion Ensemble, *Pika-don*, CD, hat(now)ART 151, 2004.
- [44] Margaret Lancaster, *io*, CD, New World Records 80665, 2009.
- [45] Erika Radermacher, Gertrud Schneider, Manfred Werder, and Tomas Bächli, *James Tenney: Bridge and Flocking*, hat ART CD 6193, 1996.
- [46] Marc Sabat and Stephen Clark, *Music for Violin & Piano*, CD, hat(now)ART 120, 1996.
- [47] James Tenney, *James Tenney: Selected Works 1961-1969*, CD, New World Records 80570, 2003.
- [48] The Barton Workshop, *James Tenney: Melody, Ergodicity and Indeterminacy*, CD, Mode 185, 2007.
- [49] ———, *James Tenney: Spectrum Pieces*, CD, New World Records 80692, 2009.