CONVERGENCE: LEVERAGING ALGORTIHMIC TECHNIQUE AGAINST THE CONSISTENCY OF MUSIC BEING MUSIC AT ALL

A Particular Approach to Speculative Music

For a composer trained in various established idioms reflecting the known configuration of musical appearances, it is difficult to consider a music that is not dependent on one’s knowledge of how, why, or even if music exists. Most composers have been trained to consider misinterpretation as a threat to compositional success. Any semiotic breakdown regarding esthesic access to the work potentially undermines that work’s content, its ability to mean anything. Our problem is deepened when we consider that any such misinterpretation, or misalignment between poietic and esthesic prerogatives, is not only possible, but inevitable. Thus the composer’s job, traditionally, is to minimize the variety of possible misinterpretations, to clarify with every greater precision compositional intention (even if unintended sound is the intention).

Our speculative approach inverts this problem and reveals it to be not merely an epistemological limitation, but rather an incompleteness concerning the material reality of that which appears. The great threat, misinterpretation, is thus ontologized: sound is not just interpreted differently by way of subjective access, but is objectively incomplete as being anything in itself. Our limited perspective, the aural finitude regarding our relation to the physical, material world around us, is a reflection of our being in an acoustic reality (of vibrational sound) before it ever aurally appears for us. There *is nothing* but the material reality of vibrational sound. This nothingness reaffirms that vibrational sound *is not all there is*; vibrational sound (or any assemblage thereof) is incomplete, ontologically un-totalizable—precisely because it ‘miraculously’ appears *for us*.

From within the contingent and thus un-totalizable acoustic reality that we find ourselves in, how does the composer begin to intervene? How does the composer (*as a material being in reality*) condition the possible *circumstances* for reality to appear to itself (other material beings) as an aural experience? This is the properly speculative compositional question. Many different composers can potentially seek to answer this question in many different ways. I propose a particular way, one that has proven useful for myself thus far and continues to raise interesting questions. I will now seek to derive its relevance to the speculative consideration of music.

Indeterminate versus Determinate Appearance

To reiterate Badiou's point: for art to take place something has to happen; material reality itself is not sufficient. Which is to say, we can't have the situation where nothing happens but the place. This is a good thing. Otherwise, the composer's job would be superfluous, and thus, truly meaningless. Ultimately, this is to say that something must intervene (happen) in the domain of aural appearances. So precisely what must happen?

For a moment, let us disregard materiality (acoustics) and consider the notion of a consistent aural appearance (such as a bird voicing the same pitched call at regular intervals for a seemingly interminable duration). Contingent necessity demands that any such consistent appearance is itself always subject to radical transformation, of becoming radically different for no reason whatsoever (the bird's call changes in some un-totalizable way or even stops being a call at all). Furthermore, if we adopt the opposite perspective, that of a chaotic aural appearance (such as the sounds of 1000 birds each with different calls), contingency still demands that such appearance could be radically different (the chaos of bird-songs is the sound of an aviary). Thus, both the aural appearance of indeterminate chaos and the appearance of determinate consistency are wholly contingent.

The necessary contingency of sonic materials and the contingent necessity governing aural appearances are both operative irregardless of any particular material or given appearance. The relevance of this appears in light of the following desire: if the composer is to intervene, then *how* (by what methodology) does she seek to specify both the materials that exist within the frame and the frame's contingent existence? Should compositional agency take the form of ever-greater specification of determinate relations? Or, should such agency proceed as an abdication of intentionality, a deferral to indeterminate chance procedures? The properly speculative answer is, of course, both—in a manner that is fully overlapping, yet never solely (completely) one or the other. My contention is that the composer should proceed in a way that modulates any ontological appearance in a way that presents nothing but the gap torn open between the two.

Modulating the Semblances

In consideration of Robert Irwin's *Scrim Veil—Black Rectangle—Natural Light* (1977), we may consider how the relations between the work's materials yielded drastically different notions of what the work was in itself. Scrim Veil was neither wholly a piece of art, nor was it nothing at all.[1] It was incomplete, precisely because its appearance shifted between the two (or three, or however many) different accounts of what it was.

[1] Of course, *Scrim Veil* has necessarily become a work of art given our perspective today. But, this necessity is nevertheless contingent, it is a product of the work having been decided upon—what we have been calling 'contingent necessity'. This process in no way undermines the immediate 'decidability' of its original presentation, or that, as being anything at all, it could have (materially) been radically different.

Our proposed response to the task of composing speculative music, as being a music that is neither wholly consistent in its presentation of sound, nor chaotic, is thus a direct reflection of the following prerogative: to modulate the work's semblances. Thus the question remains, how may consistency overlap with chaos? How may we move between the two, without either taking priority?

The Temporality of Change

We find our answer in the following way: by seeking recourse in the temporality of musical change. Change is perceived as a differential of appearances across some duration of time. In music, sounds generally appear to change through time. The differences that arise between sounds from one moment to the next can be described in mathematical ways. As Quentin Meillassoux has shown, contingent phenomena are not computable (predictable) precisely because possibility is un-totalizable. Thus, even though mathematics provides us with the distinction between contingency and chance (virtuality versus potentiality), our ability to control (intervene in) contingent processes is never fully possible; it is incomplete precisely because contingency is an absolute. Therefore, our only agency over the modulation of semblances regarding the frame of composition comes in the form of that which is totalizable and thus computable. (Even so, we must still concede that our apparent agency is itself contingent.)

It is therefore left to us to devise a particular dice-game, one that marks the apparent change of sounds through time as never wholly determined (consistent), nor completely indeterminate (random/chaotic). My particular solution to this problem (proposed dice-game) is to *deterministically change the die that we use to indeterminately change the appearance of sound*. This might seem absurdly simple, but it is an entirely sufficient means to modulate the outward appearance of what the sounds are across time in a manner that yields a wide (though not un-totalizable) set of possibilities. To generate a wide range of possibilities is desirable only insofar as the appearance of sound forces a would-be listener to question what that aural appearance is at any given moment.

Convergence of Set

We can consider the aforementioned dice-game in a theoretical way by maintaining an analogy with a real/physical die. Subsequently, I will describe some details of my own software implementation of this behavior.

Given a particular die, we identify a number of pre-given cases or potentialities corresponding to the faces of the die. When we roll the die a particular case is selected according to an (assumedly) uniform probability of selection, whereby all potential cases have equal chance of being selected. Let us arbitrarily say that we have a six-sided die, so we have six potential cases. Each case is associated with numeric values: 1, 2, 3, 4, 5, and 6, respectively. The proposed game unfolds in the following way:

1. Determine a numeric *step value* (SV) that is equal to 1 divided by some integer that is greater than or equal to 1 (for instance, 1/10 = 0.1).
2. Roll the die to determine a *selected case* (SC).
3. Record the *associated value* (AV) of the SC and store it as the *target value* (TV) for each of the die's potential cases.
4. Roll the die to determine a SC.
5. Apply the AV of the SC determined in step 4 to a parameter of sound generation.
6. Update the AV in one of the following two ways:
    i) if the TV is greater than the AV, then add the SV to the AV.
    ii) if the TV is less than the AV, then subtract the SV from the AV.
7. Change the die so that the numeric result of the previous step will be the new AV for the SC on any future rolls.

8. Go to step 4.

This procedure results in a very specific behavior. The die is initially governed by chance—the equal probability of selecting different cases (which we may refer to orthographically as: one, two, three, four, five and six). Each case is associated with different integer values (which we may define numerically as 1, 2, 3, 4, 5, and 6, respectively). However, after successive rolls, the value associated with each case progresses toward a consistent value outcome, the product of our first roll or the *target value* (TV). The more we play the game, the more the values converge toward the TV—until finally all associated values (AVs) are the same; the associated value equals the TV for all cases. To provide an example using the aforementioned die, lets say we roll the die and select five. We then set 5 as our TV. We then roll again and select six and then apply 6 to some parameter controlling sound generation. We then update 6 by *subtracting* (following step 6-ii) 0.1 from 6, which gives us 5.9. We then change the die so that the die's sixth face (case six) has an associated value of 5.9. As we continue to roll, we select cases at random (by chance) and in each instance update the values associated with the selected case and then change the die accordingly. Eventually, cases one through six all have an associated value of 5. At this point, any die-roll will yield a consistent outcome even though that very outcome was itself determined by chance. See Figure 1 for a graphical representation of value outcomes determined by 250 iterations of this exact dice-game.

If we stop here, our game is over (or it otherwise goes on for an infinite amount of time yielding the same result: 5). However, once we have *converged* we may then invert the process described in step 6 (adding or subtracting the *step value* to/from the target value), and begin to *diverge* back towards the original values. Accordingly, rather than updating the selected case's value as a means of approaching consistency, we update it in the other direction and approach randomness. See Figure 2 for a graphical representation of divergence within the bounds of the dice-game described in the preceding paragraph.

ConvergentArray: an implementation of set convergence in software

In software, the game is played using data structures rather than dice; an array of indexed values may function as a die. Our array constitutes a pre-given set of differentiated values—a set being a finite configuration of potentialities subject to probabilistic logic. To roll our die in software, we randomly select a value at a given *index*, which will replace the term *case* for the remainder of this section.

In the SuperCollider[2] programming environment, I implemented a Class Extension called "ConvergentArray" that functions like the die described above, with a few notable modifications/extensions. (See ConvergentArray.sc for the SuperCollider implementation of ConvergentArray).

Statistical Feedback Modification: The ConvergentArray object implements a statistical feedback model governing the selection of any given index in order to ensure the appearance of randomness. This model is a direct implementation of the dissonant counterpoint algorithm described by Larry Polansky, Alex Barnett, and Michael Winter in 2010.[3] (See SFRand.sc for a SuperCollider implementation of the dissonant counterpoint algorithm). True randomness, even computational pseudo-randomness, is notoriously bumpy. For our purposes, the appearance of randomness is the priority, so I

[2] SuperCollider is an audio programming environment developed by James McCartney. The software is open source and available online at the following address: http://supercollider.sourceforge.net/

[3] see Larry Polansky, et al., "A Few More Words About James Tenney: Dissonant Counterpoint and Statistical Feedback," *Journal of Mathematics and Music* 5, no. 3 (2011).

have taken pains to smooth it out: the outcomes of previous selections (history) are taken into account such that more recently selected indices are less likely to be selected and less recently selected indices are more likely to be selected. Statistical feedback biases the algorithm toward the exhaustion of the set of indices, if not series and pattern, depending on how the biasing is biased (how previously selected indices increase in their probability of selection across successive rolls).

Growth Function Modification: In the SuperCollider implementation, I further extended control over the rate and shape of convergence. The rate of convergence concerns the number of iterations (die-rolls) until all *associated values* equal the *target value*. The rate of convergence is controlled by a numerical argument that we may call the *number of steps.* The number of steps is a constant passed to each instance of ConvergentArray upon instantiation[4] that determines how many incremental additions or subtractions (steps) must occur for each initial associated value to reach the target value; fewer steps makes for faster, more abrupt convergence.

The number of steps is a critical value for computing not just the rate of convergence, but also the shape of convergence. The shape of convergence concerns the adjustability of the increment or *step value* added to, or subtracted from, the value at a given index (associated value). To refer back to our dice-game analogy, we should consider step 1 in greater detail. In step 1 we calculated a step value of 0.1 in the following way: we divided 1 (which is the smallest difference between any two values in the set of all associated values) by some integer greater than or equal to 1, for which we arbitrarily chose 10. In fact, 10 served as an arbitrary value for the number of steps to

[4] Instantiation refers to the creation of an instance of a pre-defined class or object. Here, the term signifies the creation of an instance of the ConvergentArray class.

reach the target value. We may, therefore, formalize our calculation in step 1 by providing the following generalized equation for the step value (v_s):

$$v_s = \frac{1}{N}$$

where N is a constant representing the total number of steps to reach a target value that is ±1 from the initial associated value of a given index.

In step 6 of our dice game analogy, where we update the associated value of the selected index in the direction of the target value, v_s does not change; only its sign changes (as a matter of addition or subtraction) relative to the target value. We may, therefore, consider the above equation as a parameter of the *growth function* that specifies how all associated values are to be updated. The growth function described by our dice-game analogy can be written in the following way:

$$f(x_i) = n_i \times \frac{1}{N} \times \left(\frac{T - x_i}{|T - x_i|} \right) + x_i$$

where x_i is the initial associated value of index i, T is a constant representing the target value, and n_i is the number of times index i has been selected where $0 \leq n_i \leq N(|T - x_i|)$. Essentially, we multiply the v_s by the number of times the given index has been selected (n_i). This product (either positive or negative, depending on whether T is greater than or less than x_i) is added to the value at index i (x_i). The growth function may be simplified thusly:

$$f(x_i) = \frac{n_i(T - x_i)}{N(|T - x_i|)} + x_i$$

We should notice here how the constant N does not ensure that the target value is reached in N number of steps for all associated values (x_i). In fact, N is only the actual number of steps when $T - x_i$ = -1, 1. If T = 5 and x_i = 3, then index i would need to be selected 20 times for x_i to reach T if we maintain that N = 10. To exert more control over the rate of convergence for the set of *all* associated values, we must change the growth function such that x_i for all i converge to T in N number of steps.

In the SuperCollider implementation, the growth function is changed accordingly; v_s varies proportionally with the difference between T and x_i, such that the number of times that i must be selected (n_i) for any associated value (x_i) to reach the target value (T) equals N for all i. In other words, any given index will have converged once n_i equals N. Accordingly, n_i, while necessarily greater than 0, is now bound on the upper end by N. This new growth function, the one that I have implemented in SuperCollider, looks like this:

$$f(x_i) = \frac{{n_i}^{\alpha}(T - x_i)}{N^{\alpha}} + x_i$$

where N is any integer, T is any rational number, and n_i is any integer between 0 and N inclusive. This function ensures that N establishes a universal rate of convergence, which we may define at the outset. The shape of convergence is described by the curvature of the growth function; $f(x_i)$ approaches T at a rate that is inflected by an exponential factor (α). A linear path towards convergence is defined by a power of 1 $(\alpha = 1)$, while some power greater than 1 defines an exponential path, and a power that is a fraction of 1 defines a logarithmic path. See Figures 3, 4, and 5 for graphs depicting outcome values generated using ConvergentArray with an exponential factor (α) of 1, 2, and 0.5, respectively. An array of integer values 1, 2, 3, 4, 5, 6 was used in order to provide a basis for growth function comparison with the preceding graphical representations of the dice-game analogy. All graphs converge to the value 5. This value was set artificially in order to further facilitate comparison.

Additional Modifications: It is also important to note that the ConvergentArray algorithm operates upon sets of rational numbers. Furthermore, decimals may be rounded upon output from the growth function according to a user-specified *quantization level* that is defined upon instantiation. In this way, computation proceeds with full decimal precision while allowing the user to determine if the resultant values need to be more or less exact.

All of the features implemented in the CovergentArray SuperCollider class also function in reverse, as a means to diverge the given value set. By simply counting backwards (from N to 0) the number of times a particular index is selected (n_i), a converged value set can be shown to diverge by using the same growth function. Accordingly, an infinite number of iterations (of converging and then diverging) may ensue, and an infinite number of computational modifications may be brought to bear on the parameters governing such behavior. Figure 6 provides a graph of a divergent trajectory and may be considered an inversion of the convergent trajectory shown in Figure 3.

Based on the behavior of the algorithm and the modifications discussed here, our ability to control a variety of parameters raise many questions about ‘how’, ‘when’, and ‘within what bounds’ we move to mathematically converge and diverge value sets. It is through our consideration of how the growth function changes values applied to sound synthesis that we encounter a multidimensional territory of possible change.

Using ConvergentArray to control sound synthesis parameters

Once implemented, ConvergentArray is primed to modulate the parameters of sound synthesis in a way that is neither completely predictable, nor wholly chaotic. The way that I have sought to implement such functionality is to instantiate a new array for each defined parameter governing sound synthesis. Take for example the generation of a simple sine-tone. Immediately we may want to control the sine-tone's frequency and its amplitude. My response to this situation is to instantiate two ConvergentArrays; one governs the frequency of the sine-tone, the other the amplitude. Each parameter is thereby left to converge and diverge according to its own set of values, number of steps, and growth function exponent. Furthermore, each ConvergentArray may be updated with a new set of values independently. (This is usually best to do at a point of full convergence or full divergence.)

If we imagine a sample-based instrument or instruments with dozens of parameters, with each parameter being modulated according to a ConvergentArray, the set of possible appearances of the resultant sound is vast. However, if all the ConvergentArrays are operating entirely independently, chaos remains supreme. If only the ConvergentArray governing a sound's amplitude remains consistent, while twenty other parameters vary indeterminately, the notion of consistency is itself perceptually indiscernible. Again, our goal is not merely to present chaotic change, but rather, to modulate the semblance of all sound being generated. So I have found it most effective to ensure that all parameters (or at least a high percentage of them) have converged before allowing them to diverge, and similarly, that they all should diverge before allowing them to converge. We can think of this as a gate in the algorithmically generative system that blocks all ConvergentArrays from proceeding (reversing course) until all parameters have fully realized their tasked trajectory. This ensures against disruptions in the gestalt appearance of sound as a result of ConvergentArrays falling drastically out of phase with each other.

As a result of convergent/divergent processes, aural appearances may shift in seemingly infinite ways—and not just as a matter of indeterminate selection, but rather, as the seemingly miraculous emergence and disappearance of some *telos.* As outcomes veer toward and then away from consistency (i.e. $f(x_i)$ yields T for all i), we are left with nothing but a sense of directionality that is itself wholly unpredictable and that forever seems to be lagging behind what the sounds (numerical values) are at any given moment.

Convergence in Action

Since 2009 I have implemented the ConvergentArray algorithm (as well as other convergent processes) in a number of works. However, it is only recently that I have realized how the placement of such works drastically affects what that work is. I have presented several musical experiments that use convergence of set as a principle, governing concept in concert hall settings. However, when this propositional music is presented in obvious relation to Music, it leads to nothing but misunderstanding. I have taken great pain to write out detailed program notes that mathematically describe what is going on in my pieces or (at the very least) how different sound synthesis parameters are correlated. But such efforts (my attempts to condition the context of presentation and reception) hardly serve to reframe listener’s divergent considerations when faced with my music.

However, when I thought to begin conditioning my work based on the context of its presentation (that is, addressing it ontologically rather than epistemologically), any lack of understanding regarding sound and how it was being generated no longer posed a threat. Instead, it became an opportunity to radically re-address vibrational sound's contingent status as being in relation to Music at all.

The 'Convergence' Exhibition

On September 13th, 2013, I opened a gallery exhibition featuring four new works at Stetson University's Hand Art Center in DeLand, FL. The exhibition, which featured several other works by two other faculty artists, was titled "Convergence."

Each of my four works consisted of visual and aural elements. Yet it was precisely the division between the visual and the aural in each work that functioned as leverage against any adherence to its ontological completeness.

For a more materials documenting the installation of two of my works from the "Convergence" exhibition, please visit:

1. http://www.ludicsound.com/?page=MildlySympConvo
2. http://www.ludicsound.com/?page=GivenTheMaterialsAtHand

In both of these works ConvergentArrays were used to change the appearance of sound by vacillating between consistency and chaos.

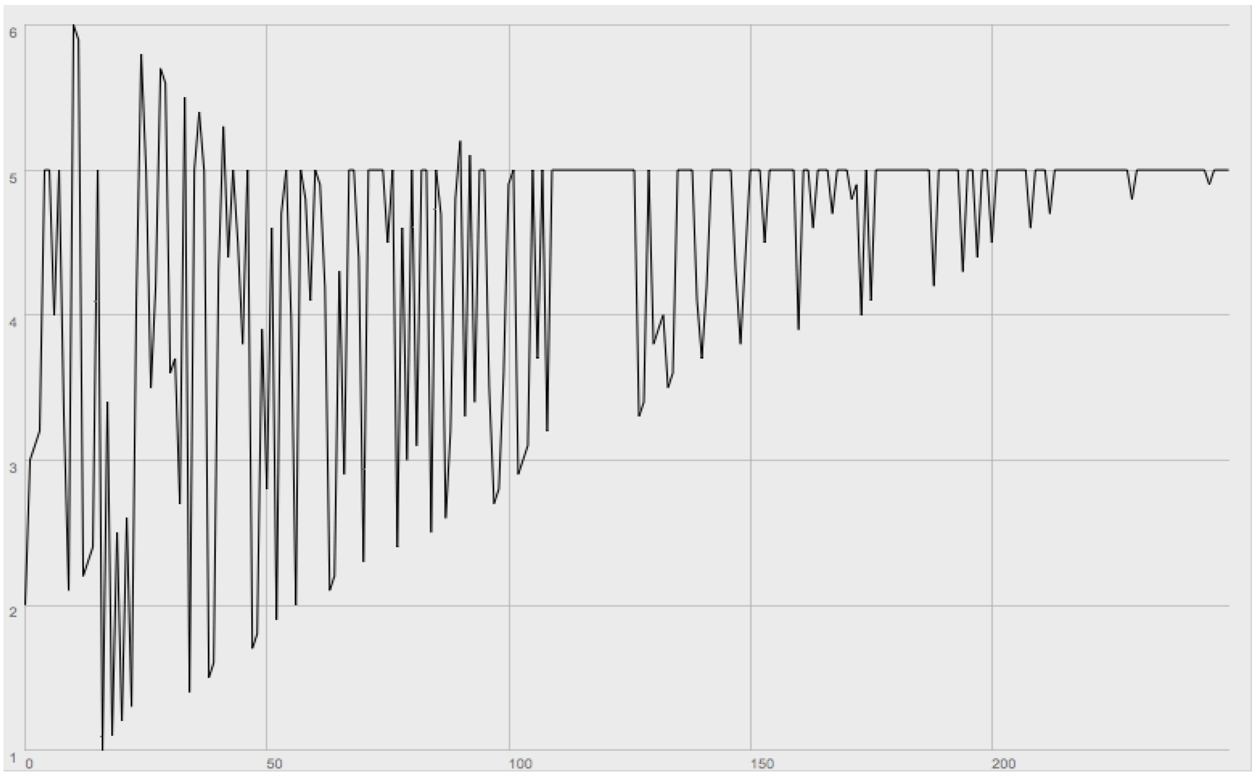

Figure 1. Convergent dice-game: associated value outcomes for 250 dice-rolls.

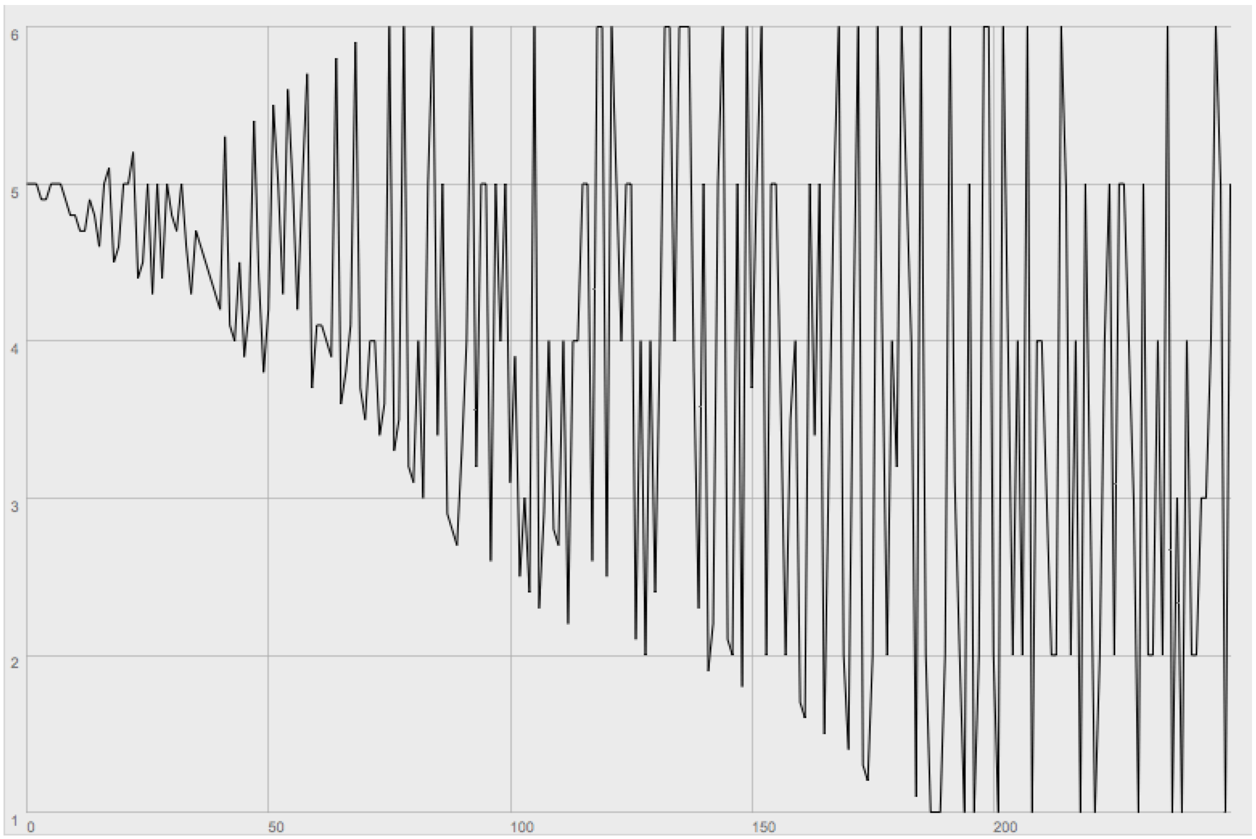

Figure 2. Divergent dice-game: associated value outcomes for 250 die-rolls.

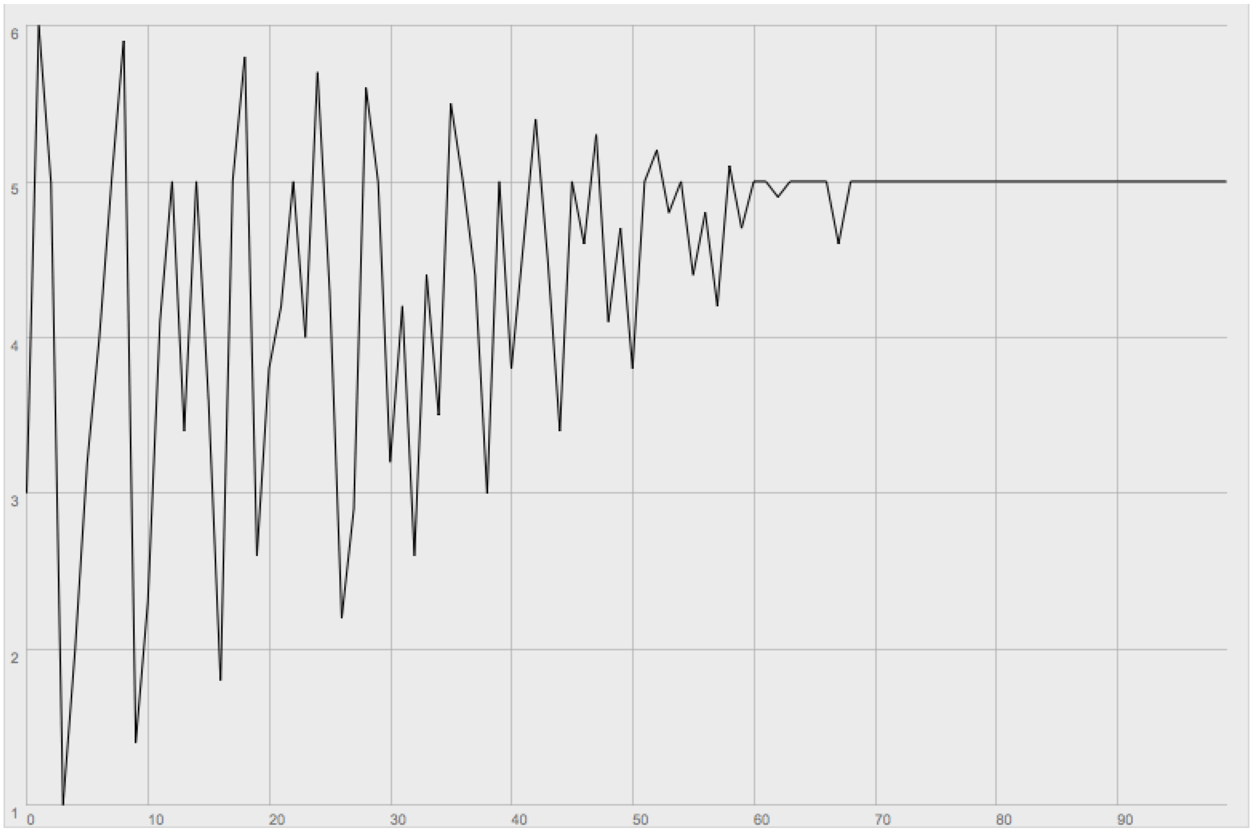

Figure 3. ConvergentArray: values [1, 2, 3, 4, 5, 6] converging across 100 iterations (N = 10, α = 1, quantization level: 0.1).

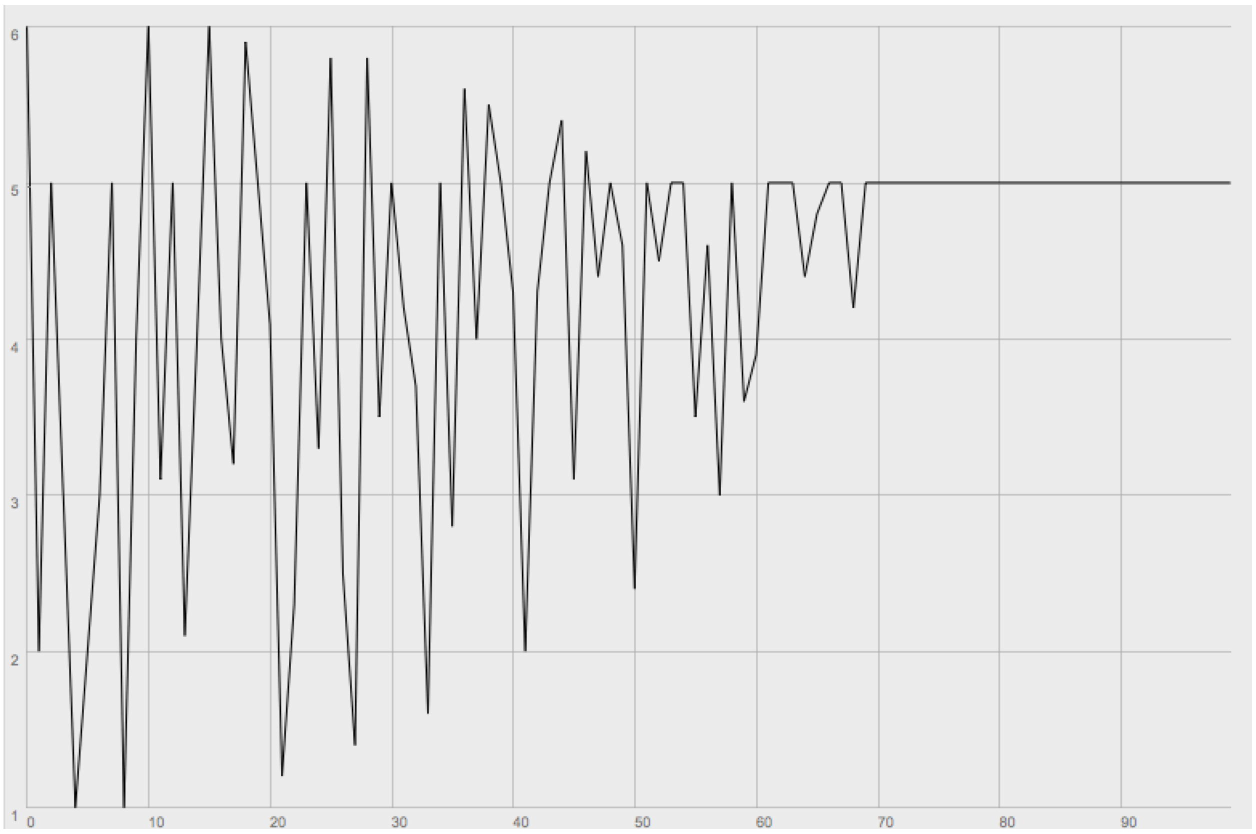

Figure 4. ConvergentArray: values [1, 2, 3, 4, 5, 6] converging across 100 iterations (N = 10, α = 2, quantization level: 0.1).

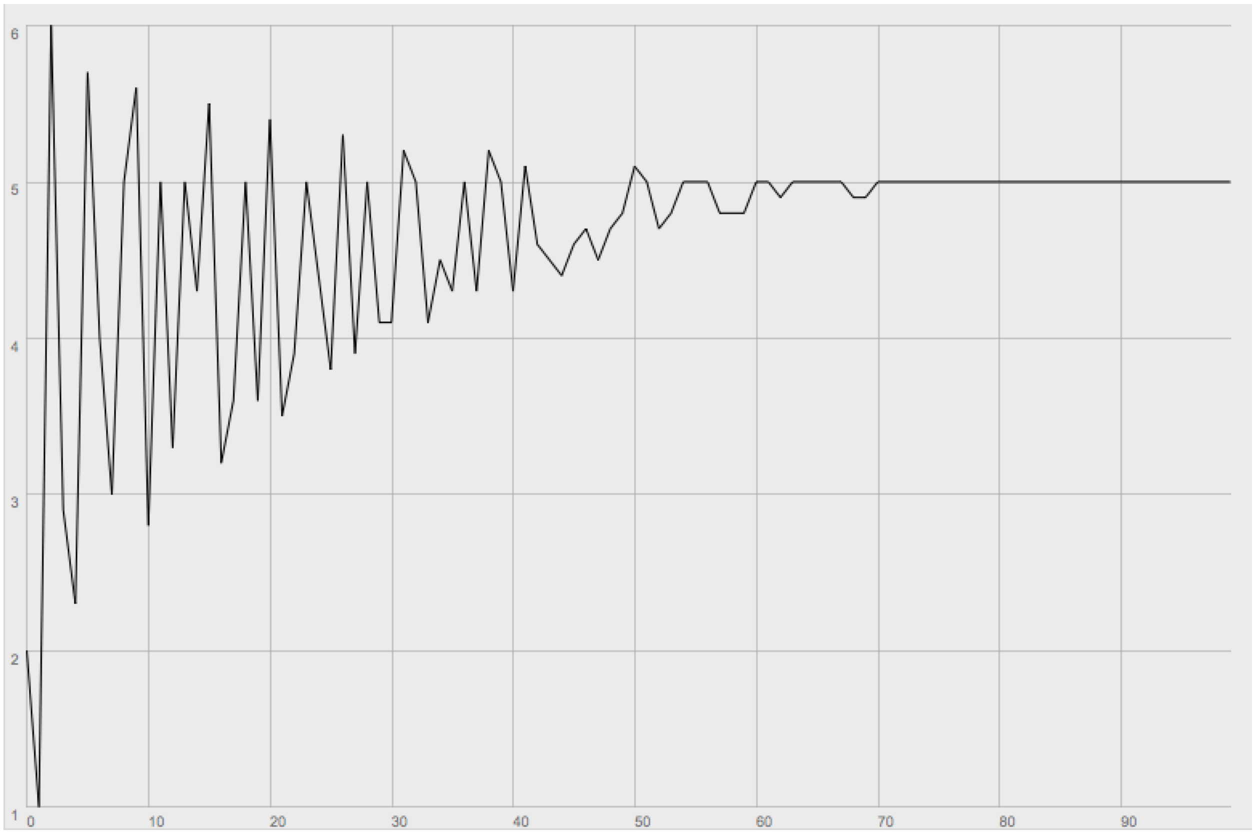

Figure 5. ConvergentArray: values [1, 2, 3, 4, 5, 6] converging across 100 iterations (N = 10, α = 0.5, quantization level: 0.1).

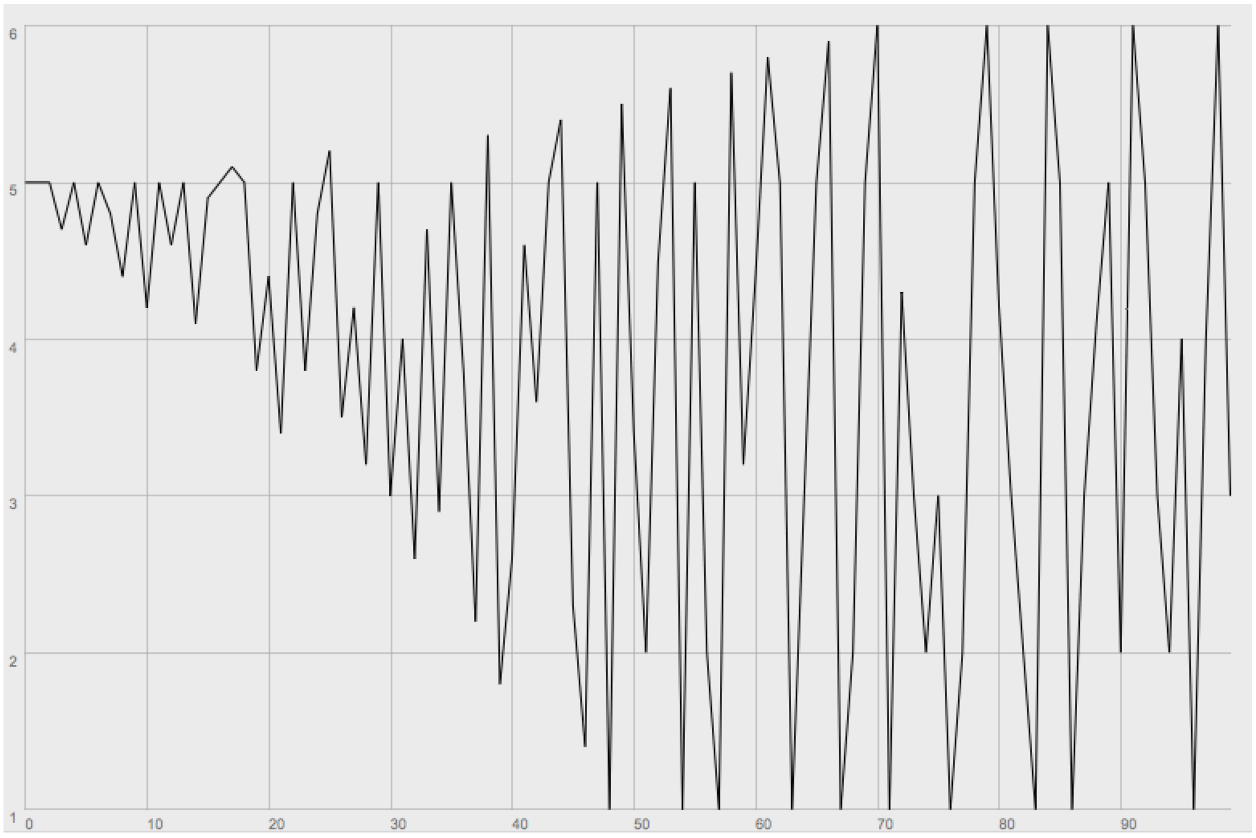

Figure 6. ConvergentArray: values [1, 2, 3, 4, 5, 6] diverging across 100 iterations (N = 10, α = 0.5, quantization level: 0.1).