The Popular Critic: Evolving Melodies with Popularity Driven Fitness

Róisín Loughran and **Michael O'Neill**
Natural Computing Research and Applications Group (NCRA)
University College Dublin, Ireland
roisin.loughran@ucd.ie

Abstract

One of the fundamental challenges in applying evolutionary computation to creative applications such as music composition is in the design of a suitable fitness function. This paper proposes a new method of examining fitness, not from an inherent musical aspect of the individual but from the degree to which a given individual conforms to the popular opinion of its peers. A cyclical system is presented that uses an initial corpus of melodies to evolve a fitness 'Critic' which in turn is used to create a new melody. This new melody is then input into the original corpus to continue the cycle of Critics creating melodies that in turn are used to create Critics. A diversity measure of the changing corpus over evolutionary cycles shows that the corpus becomes less diverse as more of the melodies are created by the system. The system creates melodies in a method that is not random but that is unpredictable to the programmer.

Introduction

This paper proposes an algorithmic music composition system based on Evolutionary Computational (EC) methods to investigate the creation of a fitness measure based on popularity within the population rather than on an inherent measure of the individual. Instead of using a numerical property belonging solely to each individual to assess its value, we attribute fitness to the individual according to how much it conforms to the dominant opinion of the population. The 'opinion' of each individual is not pre-defined but is taken as the numerical output of that individual. We do not tell the individuals what to like, we do not define what is good, we merely take a consensus of what the population chooses and evolve individuals according to how well they conform to the rest of the population. In this manner, the judgement of an individual is not random, not predetermined and not from a human observer but it is *defensible* as any computational preference should be (Cook and Colton, 2015). The final evolved individual — our *Popular Critic* — can then be used as a fitness function in a compositional run to evolve new music.

This paper describes an EC compositional system focusing on the fitness function. In aesthetic applications of EC, defining a succinct and meaningful method of measuring fitness is extremely problematic — what makes one melody 'better' than the next? Generally, this problem is addressed using a pre-determined numerical measure, a random choice or a human observer. Using a deterministic measure is not particularly creative, as taking an objective, numerical measure of how good one melody is over another is not likely to genuinely describe any aesthetic or subjective quality. Likewise, pure random choice is not creative; selecting one piece over another at random does not acknowledge any measure of merit — musical or otherwise — between them. The employment of a human observer, known as Interactive EC, is often used but must be considered less computational than autonomous methods as this is ultimately being driven by human choice. Hence, systems that employ any of these choices in their fitness functions are less computationally creative than the true definition of the term. The purpose of this system is to examine the question: what kind of measures can one use within an EC system that do not rely on predefined musical knowledge, music theory rules, similarity to given style of music or a human observer? What could a system learn to like if we did not tell it what to like?

The following section discusses previous work in the area of evolutionary music and non-deterministic fitness measurements. The remainder of the paper discusses the methods used throughout the various stages of the system, the experiments carried out and the results and conclusions drawn from these experiments.

Previous Work

This section focusses on EC methods applied to music composition. A comprehensive survey of other computational methods applied to composition is given in Fernández and Vico (2013). Details of the EC systems described can be found in Brabazon, O'Neill, and McGarraghy (2015).

Evolutionary Music

Various EC methods have been applied to the problem of algorithmic composition. Genetic Algorithms (GA) have been applied in the systems GenJam to evolve real-time jazz solos (Biles, 2013), GenNotator to manipulate musical compositions using a hierarchical grammar (Thywissen, 1999) and more recently to create four-part harmony from music theory (Göksu, Pigg, and Dixit, 2005). An adaptive memetic

search combined a GA with local search methods to investigate human virtuosity in composing with unfigured bass (Munoz et al., 2016).

Genetic Programming (GP) has been used to recursively describe binary trees as genetic representation for the evolution of musical scores. The recursive mechanism of this representation allowed the generation of expressive performances and gestures along with musical notation (Dahlstedt, 2007). Interactive Grammatical Evolution (GE) has been used for musical composition with promising results (Shao et al., 2010). GE has also been used recently with autonomous fitness functions based on statistical measures of tonality and the Zipf's distribution of musical attributes (Loughran, McDermott, and O'Neill, 2015b,a). These studies found that the representation of the music created by the grammar and the combination of individuals from the final population could be as important as the fitness function. The combination of such individuals was further explored by using a distance metric between individual segments of a melody as a fitness measure to drive the melodic evolution (Loughran, McDermott, and O'Neill, 2016).

The various attributes used in the evaluation of melodies based on pitch and rhythm measurements were discussed in de Freitas, Guimaraes, and Barbosa (2012). It was concluded that previous approaches to formalise a fitness function for melodies have not comprehensively incorporated all measures. Some studies have addressed the problematic issue of determining musical subjective fitness by removing it from the evolutionary process entirely. GenDash was an early developed autonomous composition system that used random selection to drive the evolution (Waschka II, 2007). Others used only highly fit individuals within the population from initialisation and then used the whole population to create melodies (Biles, 2013; Eigenfeldt and Pasquier, 2012).

Search Without Fitness

It has been proposed that using a pre-specified objective is not necessarily the best approach to searching. This theory suggests that searching for novelty is a better method when considering a problem, that good solutions can be found when looking for a different solution or when searching for no particular solution at all (Lehman and Stanley, 2010; Stanley and Lehman, 2015). Such a theory fits very well in searching any creative space. A musician may not know exactly what piece of music they are trying to create when they start, they work through ideas, changing their process and hence their output as they observe what they are creating. We propose that for an automated evolutionary system to be truly creative there cannot be a pre-defined objective — the fitness function should be a measure of the progress of the system.

A notable recent study demonstrated that in Computationally Creative Evolutionary systems, it is only important that the decision of fitness need be defensible; what makes one creative item better than another may not be what a human would choose but it must be a sensible, defensible and reproduceable choice by the computer program. In other words there must be a logical and explainable method in assigning fitness measures. This was investigated using the the idea of a preference function by measuring qualities such as specificity, transivity and reflexivity to determine the choice of a system in a number of subjective tasks (Cook and Colton, 2015). Such a measure may not agree with what a human may choose as the best but, most importantly, it agrees with itself. This preference function chooses one item over another due to a logical system of comparing between items and determining a decisive preference. We try to build on this idea in the system proposed.

The system and terminology proposed in this study may also be reminiscent of the evaluation framework proposed in Pearce and Wiggins (2007). The proposed study differs in a number of important ways. This study does not attempt to conform to any particular style or genre of music but instead attempts to create an opinion among naive agents or 'Critics'. No indication as to whether the original melodies are good or bad is given. Furthermore, the proposed system is cyclical in nature, whereby the output is input back into the system for a dynamic evolution of further critics. Finally we do not include human evaluation or discrimination tests in our evaluation of the results, but instead focus on the diversity of the melodies produced. There is no aim towards human mimicry or trickery within this system.

The consensus of the population idea proposed here also shares conceptual similarities with the method in Miranda (2003), which co-evolved agents with repertoires of melodies according to a measured 'sociability'. This sociability was measured in terms of similarity of the agent's repertoires; individual melodies survived or were altered depending on reinforcement feedback between co-evolving agents. This fitness differs from our proposed method as it is the correlation of a individual's opinion to that of the (single) population that is measured in this system rather than a direct similarity measure between melodies.

Contribution of this Work

Earlier versions of the proposed system studied the musical representation resultant from the grammar used and the effect of the combination of the individuals in the final population more so than examining the fitness measures driving the evolution (Loughran, McDermott, and O'Neill, 2015b,a).

The proposed paper is intended to extend this work further into the 'meta' domain of musical creation by attempting to discover new and autonomous ways of driving EC systems with novel ideas for fitness measures. The objective of this study is not to create 'better' music but to observe the behaviour of melody creation when the fitness function is dynamically evolved to be *Popular*. Our 'Popular Critic' is adjudicated as to how much it agrees with the population rather than any inherent quality within itself. The term Critic was chosen to convey the sense of adjudication or preference implied by its output. It is proposed that this method mirrors the typical social exposure of music in the real world; whether we like to admit it or not, we are exposed to certain music more than others throughout the course of our lives and this exposure has a profound effect on our musical taste. If we choose to go with the crowd or against it, musical exposure and our perception of what is popular influences the music we choose to listen to, resulting in a cyclical system

of further exposure and judgement. This paper proposes a framework to mimic this cycle through an evolutionary system of creating Critics that agree with the popular choice of music that in turn are used to evolve new music that in turn is used to create new Critics.

Method

There are three distinct phases to this compositional system:

- The evolution of an initial musical corpus using GE
- The evolution of a Critic that conforms to the population's opinion as to which are the best melodies
- The evolution of novel music using this evolved Critic as a fitness measure which then replaces one of the original melodies in the corpus

As the method is heavily based on GE, a brief introduction is given below.

Grammatical Evolution

GE is a grammar based algorithm based on Darwin's theory of evolution. As with other evolutionary algorithms, the benefit of GE as a search process results from its operation on a population of solutions rather than a single solution. From an initial population of random genotypes, GE performs a series of operations such as selection, mutation and crossover over a number of generations to search for the optimal solution to a given problem. A grammar is used to map each genotype to a phenotype that can represent the specified problem. The success or 'fitness' of each individual can be assessed as a measure of how well this phenotype solves the problem. Successful or highly fit individuals reproduce and survive to successive generations while weaker individuals are weaned out. Such grammar-based generative methods can be particularly suitable to generating music as it is an integer genome that is being manipulated rather than the music itself. This allows the method to generate an output with a level of complexity far greater than the original input. This added complexity generation is helpful in creating interesting and diverse pieces of music. In the system proposed, the grammar defines the search domain — the allowed notes and musical events in each composition. Successful melodies are then chosen by traversing this search space according to the defined fitness function.

We exploit the representational capabilities of GE resulting from the design of a grammar that defines the given search domain. GE maps the genotype to a phenotype — typically some form of program code. This phenotype can then be interpreted by the user in a predetermined manner. In these experiments, the programs created are written in a command language based on integer strings to represent sequences of MIDI notes. We design a grammar to create this command language which is in turn used to play music. An overview of the GE process including the mapping of the grammar to MIDI notes is shown in Figure 1.

Experimental Setup

This section describes the implementation of GE in all phases of the system. A graphical overview of the system depicting the flow between the phases of the process is shown in Figure 2.

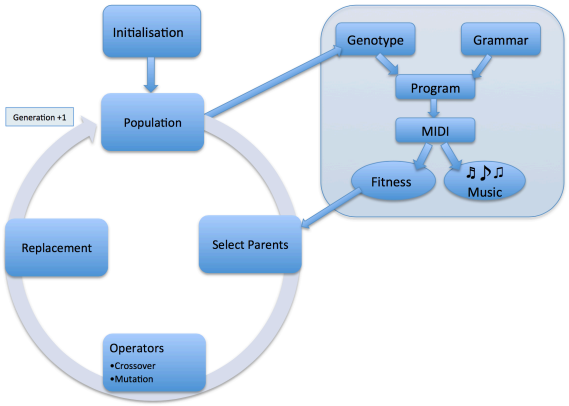

Figure 1: Overview of Grammatical Evolution.

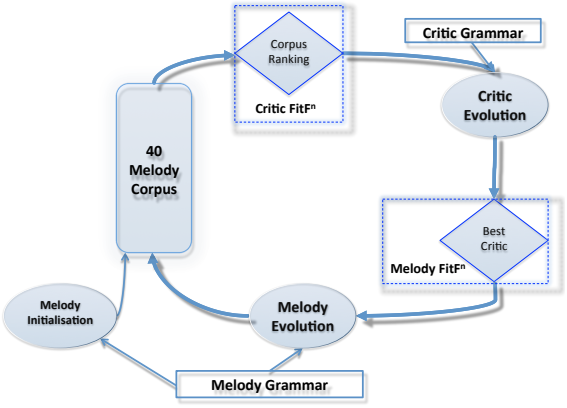

Figure 2: Flow diagram of the Popular Critic System.

Creating the Musical Corpus

Our Popular Critic is evolved according to its agreement with a population of its peers on their opinion of a selection of melodies. At initialisation, an initial corpus of 40 MIDI melodies was created using a previously developed system for composing short melodies with GE. A full description of this method and the results obtained can be found in Loughran, McDermott, and O'Neill (2015b). The following is an overview of the grammar and fitness measure used in the system. The grammar used is based on:

```
<piece>::=<event>|<piece><event>
 |<piece><event><event>
 |<piece><event><event><event>
<event>::=<style>,<oct>,<pitch>,<dur>
<style>::=<note>|<note>|<note>|<note>
 |<note>|<note>|<note>|<note>
 |<chord>|<chord>|<chord>
 |<chord>|<turn>|<arp>
<turn>::=<dir>,<len>,<dir>,<len>,<stepD>
<len>::=<step>|<step>,<step>
 |<step>,<step>,<step>
```

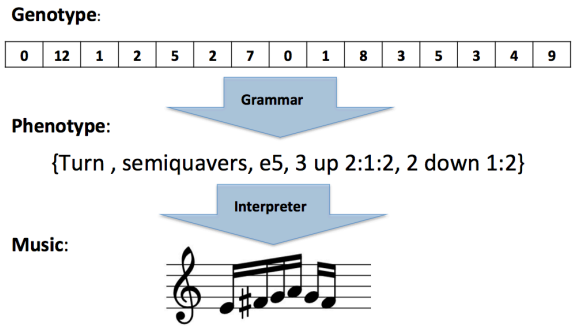

Figure 3: Application of Melody Grammar to integer Genotype through to representational Phenotype that can be interpreted into music.

```
 |<step>,<step>,<step>,<step>
<dir>::=down|up
<step>::=1|1|1|1|2|2|2|2|2|2|2|3
<stepD>::=1|2|2|2|2|2|2|4|4|4|4|4|4
<oct>::=3|4|4|4|4|5|5|5|5|6|6
<pitch>::=0|1|2|3|4|5|6|7|8|9|10|11
<dur>::=1|1|1|2|2|2|4|4|4|8|8|16|16|32
```

This grammar creates a melody `<piece>` containing a number of notes with specified pitch and duration. Each `<event>` can either be a single note, a chord, a turn or an arpeggio. A single note is described by a given pitch, duration and octave value. A chord is given these values but also either one, two or three notes played above the given note at specified intervals. A turn results in a series of notes proceeding in the direction up or down or a combination of both. Each step in a turn is limited to either one, two or three semitones. An arpeggio is similar to a turn except it allows larger intervals and longer durations. The application of this grammar results in a series of notes each with a given pitch and duration. The inclusion of turns and arpeggios allows a variation in the number of notes that are played, depending on the production rules chosen by the grammar.

This grammar is combined with the genotype to create the given phenotype — which can now be interpreted into MIDI note values. An example of this genotype to phenotype mapping for a short phrase is shown in Figure 3. This illustrates how a series of integer values can be transformed and interpreted in to a series of notes of specified pitch and duration. The selection of melodies into future generations is based on the defined fitness function. For this initial corpus the fitness is taken as a measure of the length of the melody combined with a statistical measure of prevalent tones within the piece. This is used to encourage the emergence of a pseudo-tonality (in that numerous pitches are repeated more often than others) but it does not enforce a key signature on any of the melodies. Initially the fitness is measured as:

$$fitness_{initial} = (Len - 200)^2 + 1 \quad (1)$$

where Len is the length of the current phenotype.

For an emergent tonality one pitch should be the most frequently played within the melody, with an unequal distribution of the remaining pitches. In the fitness the *primary* is defined as the pitch value with most instances and the *secondary* as that with the second highest number of instances. Thus for a good (low) fitness the number of primary pitches must be significantly higher than the number of secondary pitches. Furthermore, the number of instances of the seven most frequently played notes as Top7 and the number of instances of the top nine notes as Top9.

If any of the following inequalities hold:

$$\frac{\#\text{ instances of primary}}{\#\text{ instances of secondary}} < 1.3 \quad (2)$$

$$\frac{\text{Top7}}{\text{Total number of played notes}} < 0.75 \quad (3)$$

$$\frac{\text{Top9}}{\text{Total number of played notes}} < 0.95 \quad (4)$$

the fitness is multiplied by 1.3. This enforces the primary tone to have significantly more instances than the secondary and encourages most of the notes played to be within the top seven or top nine notes. These limits of 0.75 and 0.95 enforce more tonality than 12 tone serialism but will not create a melody with typical Western tonality. For these experiments, the top four melodies in the final population are concatenated together to encourage the emergence of themes within the final compositions. This grammar and fitness function create the corpus of 40 MIDI melody compositions which is then used to evolve the musical *Critics*.

Evolving the Critic

The purpose of this experiment is to dynamically design a new fitness function for adjudicating melodies that is not known to the programmer at the outset of the experiment. Our Critic is evolved to become the fitness measure to adjudicate the evolution of future melodies. This Critic (i.e. the fitness function) is itself evolved in the second phase of the experiment. GE is used to create this Critic as a specified linear combination of the content of the melodies.

The 'Popular Critic' is evolved by creating a population of individuals (or Critics), each of which gives a numerical 'opinion' of each of the melodies in the corpus. The melodies are represented as the number of times each degree of the scale and each note duration is played within the melody. Thus every melody is reduced to a list of 18 integer values. These instances are incorporated with a new grammar in GE shown below:

```
<expr> ::= <O><T1><O><T2><O><T3><O><T4>
   <O><T5><O><T6><O><T7><O><T8><O><T9>
   <O><T10><O><T11><O><T12><O><D1><O>
   <D2><O><D4><O><D8><O><D16><O><D32>
<O> ::= <op><scalar>
<op> ::= + | - | *
<scalar> ::= 1 | 2 | 3 | 4 | 5
```

This very simple grammar takes each of the 12 tonal and 6 duration instances, multiplies each by a value 1-5 and then either adds, subtracts or multiplies it by the previous values. This outputs a scalar value resulting from a linear combination of the 18 given values. Each individual in the population results in a numerical value for each of the 40 given

Table 1: Scores and Rankings of each piece by each person

Scores	M	T	J	R	U2
Ann	-2	4	5	3	7
Barry	-3	5	2	0	3
Ciara	-1	3	4	1	0
Donal	2	10	4	0	-50
Rankings	M	T	J	R	U2
Ann	1	3	4	3	5
Barry	1	5	3	2	4
Ciara	1	4	5	3	2
Donal	3	5	4	2	1

melodies. This is currently a *meaningless adjudication* of the melody — there is nothing to say that 11 is better than 5 — it is merely a unitless numerical assignment.

In this experiment, however, we attribute 'preference' to this numerical output. The melodies are ranked 1-40 according to this numerical value, calculated by the given individual (the current Critic). These rankings are averaged across all individuals in the population and the overall ranking of the melodies across the population (of all Critics) is found. This overall ranking of all 40 melodies is taken as the popularity consensus of the population. The fitness of each individual Critic is then calculated according to how closely it correlates with this overall popularity, hence the fitness of the individual Critic is aligned with how much it conforms to the consensus of the population of Critics. The Kendall-Rank Correlation is used to calculate this fitness. Selection, Crossover and Mutation are then performed over successive generations to evolve one best 'Popular Critic' as with typical EC methods. The best evolved Popular Critic is saved to be used to evolve new music in the final phase of the system.

This section is the crux of the proposed experiment. To illustrate the workings of this section, consider the following scenario: Ann, Barry, Ciara and Donal are all given 5 pieces of music to listen to — a Melody (M), Tune (T), Jig (J), Reel (R) and a U2 song (U2). They are asked to give a numerical value of how much they like each piece — the larger the number, the more they liked it. We note the ranking of their choices. These scores and calculated rankings are shown in Table 1. The absolute values of the scores does not matter, only the overall ranking of the 5 is noted. From this we calculate the preferred ranking across the four people for all the music [M T J R U2] as [1 5 4 2 3]. We measure the fitness of each person as how close their own choice correlates with this overall opinion — finding that Barry's choices come closest. In this scenario Barry would be chosen as our Popular Critic.

Creating New Music with the Critic

The best evolved Critic can be used as the fitness function in a new GE run to evolve new music. The grammar used is similar to that used to create the original corpus. As before, in each generation a population of melodies is created from this grammar. The fitness of each melody is measured as the numerical output of the Critic on the given melody. A minimising fitness function is used resulting in melodies with smaller Critic outputs being favoured for selection over those with higher outputs.

Table 2: EC parameters common to each evolutionary phase

Parameter	Value
Population Size	100
No. Generations	50
Selection	Tournament (size 2)
Crossover Rate	0.7
Mutation Rate	0.01
Initial Genome Length	100
Elite Size	1

As explained above, each Critic is comprised of a combination of the 18 instance values. Each instance can be combined using the prefix `+`, `-` or `*`. This means that the range of output results for the evolved melody can vary widely depending on the Critic used. A Critic that contains a large number of negative terms will result in more small, negative outputs for melody fitness than one with mostly positive terms. In the extreme, a Critic with mostly negative terms combined with multiplication terms can result in highly negative values whereas one with purely positive terms will minimise to zero. Thus, certain Critics can result in extreme negative results by increasing the length of the melody, resulting in an unwanted bias towards longer melodies for these Critics. To combat this, the length of the melody is controlled within the fitness calculation. If the melody is less than 100 in measured duration, the fitness is calculated as the output of the Critic applied to that given melody. If the melody is longer than this the fitness is:

$$\text{fitness} = \text{Critic(melody)} + (Len - 100)^2 \qquad (5)$$

This results in a heavy penalty on longer compositions. To prevent overly long compositions escaping this inequality, an added constraint of minimum fitness is added to this phase. Thus the evolutionary run stops after either the specified number of generations has passed or if the fitness reaches -1000.

This method is used to repopulate the original corpus of melodies with melodies created by the system. After a new melody has been created it replaces one original melody from the corpus and the entire process is run again. When this is repeated 40 times, the original corpus has been replaced by melodies created by the system. In this way, the system loops around by itself, creating new melodies from Critics that have learned from the previous output of the system. Each of the evolutionary phases within the experiment were run with the common parameters shown in Table 2.

Results

This section describes the outputs of the phases of the system. A selection of the final created melodies can be found at `http://ncra.ucd.ie/Site/loughranr/music.html`.

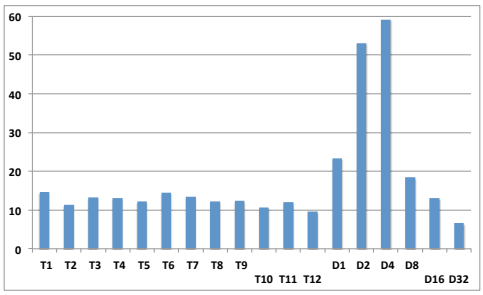

Figure 4: Average number of instances of each of the 18 tonal and duration values in the 40 initial melodies.

Initial Corpus

As detailed above, each melody is represented as a linear combination of the number of instances of each degree of the scale and each duration of note within the melody. Figure 4 displays the average value for each of these measures across the initial corpus. This shows that by far the two most prevalent values are the number of semiquavers (D2) and the number of quavers (D4) in the melodies. This is unsurprising as due to the grammar used, several of these notes are introduced every time a run or arpeggio is played resulting in these being the most common duration values. There appears to be no bias towards specific pitch values. Again, this is as expected as no key signature has been specified and the grammar did not favour any pitch over any other.

Evolving the Critic

Fitness Evolution For any evolutionary run to be deemed successful, the best achieved fitness must improve over successive generations. To investigate this, the Critic Evolution phase of the system was run 40 times (independently) and the average improvement of the system over successive generations was noted. A plot of this fitness improvement is shown in Figure 5. As can be seen, both the best and average fitness display a dramatic improvement in the first 10 generations. This improvement gradually tapers off in the following 10 generations and remains approximately stable thereafter. As described earlier this fitness is taken as a measure of the correlation between the individual and the most popular opinion of the overall population. Over successive generations, we would expect the best fitness to improve as the population converges on a 'most popular' vote and one individual manages to approximate it. Hence, as expected the best and average fitness is seen to improve, but as the crossover and mutation operators are used until the final generation the population does not converge completely.

Diversity Among the Critics Each 'best' Critic is evolved over 50 generations in accordance to how well it agrees with the population in its judgement of the corpus. As the corpus of melodies does not change over generations, we would expect the population of Critics to develop some similarities over generations as the critics begin to converge on what they agrees to be 'good' melodies. This hypothesis was tested by examining the diversity of the Critics over a series

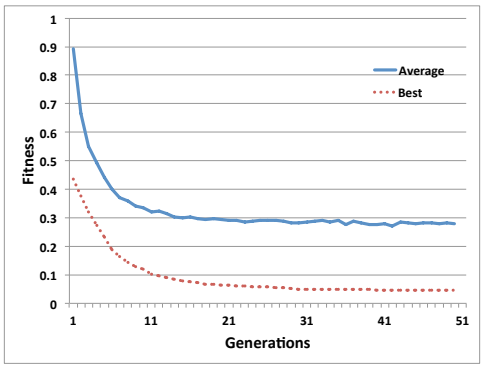

Figure 5: Average and best fitness over 50 generations averaged across 40 independent evolutionary runs.

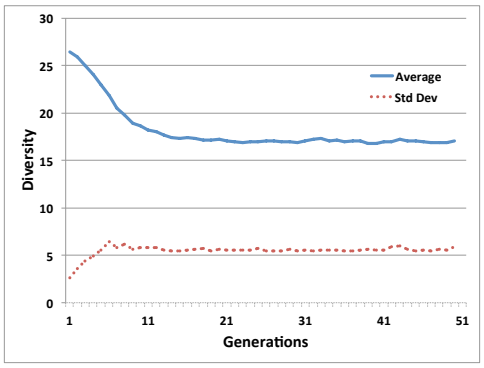

Figure 6: Average diversity of the Critics evolved over 50 generations across 40 independent evolutionary runs.

of independent runs. The population diversity was measured as the sum of the Levenshtein edit distance between the phenotypes of each pair of Critics. A plot of the average and standard deviation of this diversity measure averaged across all 40 runs is shown in Figure 6. This shows a marked decrease in the average diversity in the first 10 generations. This reduction in diversity correlates with the observed decrease in fitness displayed in Figure 5. This demonstrates that, as expected, a decrease in fitness results in a corresponding decrease in diversity in the population. As the population converges, the better Critics move towards a general consensus in their 'opinion' of the corpus of music.

Melody Creation

Once the best Critic has been evolved, it can be used in a further evolutionary run to create new music. For these experiments the Critic is used as a minimising fitness function; melodies that result in a smaller output from the given Critic are deemed 'more fit' than those with a higher resultant output. This is an arbitrary choice, maximising or evolving towards a constant could be used instead, but minimising was chosen in keeping with the minimising of the fitness results in earlier phases of the system. The grammar used is the same as that used to create the original corpus. A population of 100 melodies is evolved, the best four of which are com-

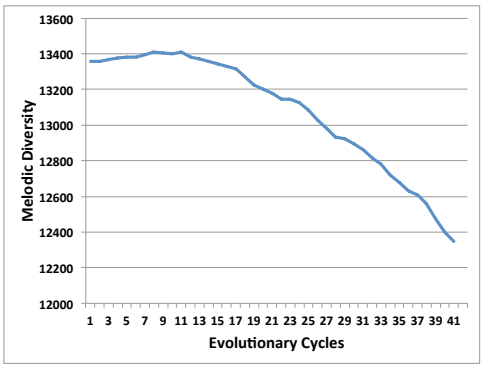

Figure 7: Diversity within the corpus as the Critic replaces the melodies over 40 evolutionary runs.

bined to create the resultant melody. At the end of each run, this resultant best melody replaces a melody in the original corpus and the cycle starts again.

When this cycle is repeated 40 times, the initial corpus of melodies has been completely replaced by melodies created by the system. To consider the change in the corpus over the course of the run, we compared the diversity of the melodies present in the corpus as the corpus was re-filled by evolved melodies. The diversity of the given corpus was measured as the sum of the Levenshtein distances between the representation of each pair of melodies in the corpus. The change in this diversity across 40 runs is shown in Figure 7. This shows that for approximately 10 evolutionary cycles, the diversity does not change dramatically from that of the initial corpus, but after 15 cycles, as the corpus is filled by newly evolved melodies there is a steady decrease in this measured diversity. The decrease is small, but nevertheless displays a definite trend. This shows that the process is having a directed effect on the melodies being produced. The original corpus was created without any preference from a Critic. This diversity reduction shows a move towards similarity in melodies created by the evolved Critics.

Discussion

As stated from the outset, this system was not created directly to produce 'better' melodies, but to look to ways of encouraging an autonomous system to determine its own preference for one melody over another. We must again stress that the individual Popular Critic has not been evolved to make a specific knowledgable *judgement* of a melody. The numerical value of the Critic as it itself is being evolved does not have meaning on its own, it only has merit in relation to the numerical outputs of all other Critics in the population. This system cannot and does not evolve music in any known 'human' musical way; the purpose of the system is to investigate other ways of adjudication.

Such a system is a way of searching for new ideas in an unpredictable way, as in the stepping stone method proposed in the myth of the objective (Stanley and Lehman, 2015). In listening to the melodies, we can hear aspects of the grammar such as runs, arpeggios, chords and single notes. The repetition of themes within the compositions indicate that the top individuals in the final population are similar (but not identical). This indicates that the evolved Critics are capable of traversing the melodic search space to converge on a good idea. From the selection of melodies, it is evident that the system is capable of creating a wide variety of melodies. Melody16 for example is full of fast runs whereas Melody37 contains barely any notes at all. This is because the number of notes was not specified at any time during the process, the genotype-phenotype mapping created by the grammar provides a rich musical domain in which the system can search.

This idea of evolving music according to popularity can somewhat crudely mirror the manner in which music is subject to popular opinion in real life. Popularity is not always predictable, it is formed from the consensus of many different individuals. Popularity is not necessarily an objective measure of goodness or merit. This is true for this system as it is true in the real world; just because a particular melody happens to have a good numerical measure from more Critics than another does not mean it would be deemed better by a human. Music appreciation in general can suffer from similar problems however. We may like to say we decide ourselves as to music we like and don't like, it is naive to state that our cumulative exposure to music over the years has not had a direct effect (negative or positive) on any such choices we make. We propose this system may explore and exploit this type of exposure we are subjected to as a society.

In recent years, a number of tests have been proposed to evaluate creative systems (Ariza, 2009). One such measure used to test for the presence of creativity is the Lovelace Test (Bringsjord, Bello, and Ferrucci, 2003). This test states that in order for creativity to be present the output of a system must be one which the programmer could not explain or predict. The grammar used in this system may dictate the search space available for composing these melodies but in no way can the programmer predict what melodies will be composed by this system, or even which tone or duration instances will be most important for this decision. Generally, the fitness function is what drives any evolutionary process, but in this system the programmer does not directly control the creation of this fitness function either — it is created as a process of population convergence or social agreement. The programmer does not control what is good or what survives to future generations. In this way, we feel that this system brings the application of EC methods in aesthetic domains such as music a step closer to true computational creativity.

Admittedly, at the moment, the evolution of the Critics is not entirely independent from human defined judgement; the Critics are originally asked to judge melodies that were themselves created by a human-defined fitness function. Changing the original corpus (or the fitness function used in creating it) will have an effect on the final result, so it would be erroneous of us to claim that this system is completely autonomous and free from human interference at this point. It is difficult to get the workings of the system to decouple from this initialisation (or 'Creation'). After initialisation however, the system runs without any form of human judgement, replacing the originally created melodies. We hope to develop this system towards complete autonomy

— if we could observe autonomous creation not just in the cycling of the system but in the initialisation we could investigate the possibilities of observing true artificial creation.

Conclusion

This paper describes a new system for using EC methods to evolve melodies by creating a fitness function based on the popularity of a population of critics. This method does not try to define a numerical measure of what is aesthetically 'good' but merely proposes that popularity, consensus or agreement among a population of generated individuals can be used to drive an evolutionary system to create new melodies. Such melodies are not randomly generated, but are also not predictable from the outset of the experiment.

We believe that there are many exciting possibilities in developing this framework. At the moment, the melodies possible from the system are heavily dependent on and constricted by the grammar used. Furthermore, the representation of the melodies is very constricting in the measurement of the melodies. Our next immediate step is to use this system to create more interesting and appealing music while expanding this representation to depict more meaning within the melodies produced. We would like to develop this system to be able to represent and learn from 'real-world' melodies. Could a development of this system learn to appreciate and therefore reproduce style? If we populated the initial corpus with lullabies, could it reproduce a new lullaby? At the extreme, if we could populate the corpus with the top 40 pop songs could we evolve a Popular Critic to quantify this popularity in such a way as to produce a new successful pop song? This may seem far-fetched at the moment but we hope that evolutionary creative methods such as this will help develop towards such a system.

Acknowledgments

This work is part of the App'Ed project funded by Science Foundation Ireland under grant 13/IA/1850.

References

Ariza, C. 2009. The interrogator as critic: The turing test and the evaluation of generative music systems. *Computer Music Journal* 33(2):48–70.

Biles, J. A. 2013. Straight-ahead jazz with GenJam: A quick demonstration. In *MUME 2013 Workshop*.

Brabazon, A.; O'Neill, M.; and McGarraghy, S. 2015. Grammatical evolution. In *Natural Computing Algorithms*. Springer. 357–373.

Bringsjord, S.; Bello, P.; and Ferrucci, D. 2003. Creativity, the turing test, and the (better) lovelace test. In *The Turing Test*. Springer. 215–239.

Cook, M., and Colton, S. 2015. Generating code for expressing simple preferences: Moving on from hardcoding and randomness. In *Proceedings of the Sixth International Conference on Computational Creativity June*, 8.

Dahlstedt, P. 2007. Autonomous evolution of complete piano pieces and performances. In *Proceedings of Music AL Workshop*. Citeseer.

de Freitas, A. R.; Guimaraes, F. G.; and Barbosa, R. V. 2012. Ideas in automatic evaluation methods for melodies in algorithmic composition. In *Sound and Music Computing Conference*.

Eigenfeldt, A., and Pasquier, P. 2012. Populations of populations: composing with multiple evolutionary algorithms. In *Evolutionary and Biologically Inspired Music, Sound, Art and Design*. Springer. 72–83.

Fernández, J. D., and Vico, F. 2013. AI methods in algorithmic composition: A comprehensive survey. *Journal of Artificial Intelligence Research* 48:513–582.

Göksu, H.; Pigg, P.; and Dixit, V. 2005. Music composition using genetic algorithms (GA) and multilayer perceptrons (MLP). In *Advances in Natural Computation*. Springer. 1242–1250.

Lehman, J., and Stanley, K. O. 2010. Efficiently evolving programs through the search for novelty. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, 837–844. ACM.

Loughran, R.; McDermott, J.; and O'Neill, M. 2015a. Grammatical evolution with zipf's law based fitness for melodic composition. In *Sound and Music Computing Conference, Maynooth*.

Loughran, R.; McDermott, J.; and O'Neill, M. 2015b. Tonality driven piano compositions with grammatical evolution. In *Evolutionary Computation (CEC), 2015 IEEE Congress on*, 2168–2175. IEEE.

Loughran, R.; McDermott, J.; and O'Neill, M. 2016. Grammatical music composition with dissimilarity driven hill climbing. In *Evolutionary and Biologically Inspired Music, Sound, Art and Design*. Springer.

Miranda, E. R. 2003. On the evolution of music in a society of self-taught digital creatures. *Digital Creativity* 14(1):29–42.

Munoz, E.; Cadenas, J.; Ong, Y. S.; and Acampora, G. 2016. Memetic music composition. *IEEE Transactions on Evolutionary Computation* 20(1).

Pearce, M. T., and Wiggins, G. A. 2007. Evaluating cognitive models of musical composition. In *Proceedings of the 4th international joint workshop on computational creativity*, 73–80. Goldsmiths, University of London.

Shao, J.; McDermott, J.; O'Neill, M.; and Brabazon, A. 2010. Jive: A generative, interactive, virtual, evolutionary music system. In *Applications of Evolutionary Computation*. Springer. 341–350.

Stanley, K. O., and Lehman, J. 2015. *Why Greatness Cannot Be Planned: The Myth of the Objective*. Springer.

Thywissen, K. 1999. GeNotator: an environment for exploring the application of evolutionary techniques in computer-assisted composition. *Organised Sound* 4(02):127–133.

Waschka II, R. 2007. Composing with genetic algorithms: GenDash. In *Evolutionary Computer Music*. Springer. 117–136.