In *Proceedings of the 2001 International Joint Conference on Neural Networks*
(IJCNN-01, Washington, DC). IEEE, 2001.

Creating Melodies with Evolving Recurrent Neural Networks

Chun-Chi J. Chen (ccchen@cs.utexas.edu)
Risto Miikkulainen (risto@cs.utexas.edu)

Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712

Abstract

Music composition is a domain well-suited for evolutionary reinforcement learning. Instead of applying explicit composition rules, a neural network is used to generate melodies. An evolutionary algorithm is used to find a neural network that maximizes the chance of generating good melodies. Composition rules on tonality and rhythm are used as a fitness function for the evolution. We observe that the model learns to generate melodies according to these rules with interesting variations.

1 Introduction

Since Leonard Bernstein's lecture on music and language in 1973 at Harvard, many music scholars have worked on the theory and practice of *musical grammar* [4]. The idea is to draw comparisons of music and language in terms of syntax, grammar, and semantics. For example, Lerdahl and Jackendoff [7] formulated a theory of musical grammar to analyze music in a syntactic way. In contrast to other composers, Lerdahl and Jackendoff followed strict rules in musical grammar and composed melodies in the same way as people formulate sentences in a language.

Following this idea, in this paper we assume that music is a form of language that can be expressed with musical grammar, and rules in such grammar are numerous and universal. Since the grammar is large, we hypothesize that different composers utilize different subsets of these rules. Bela Bartok, a well-known 20th century composer, can be used to illustrate this hypothesis. Bartok's work has been analyzed in detail; for example, Antokoletz [2] proposed a system that could serve as a means to formulate and organize Bartok's music. One characteristic of Bartok's work is the use of certain cell structures (patterns of tonal music), which are found more frequently in his work than those of other composers [3]. Based on this analysis, we may design a system to simulate Bartok's music by using a common set of musical grammar rules and a specific set of rules about the cell structures characteristic to Bartok. We believe that the results from this system will sound, to an extent, like Bartok's music.

Artificial composition has been an active research area for a long time. Techniques involving perception [8], emotion [13] and psycho-acoustics [11] have been proposed. These approaches presumed a certain state of mind and tried to approximate the stream of thought while a composer is writing his or her music. While the results were promising, they often lacked flexibility in generating variations in the melody, and they drew little support from music theory. Other approaches were more scientific in that they used a statistical model [11] or a knowledge base [1] to estimate or predict each step in the composition. These models worked well in emulating specific style of composition, but they were not able to generate variation not predefined in the database. Recently, Todd and Loy [15] and Desain [5] used a connectionist approach on music composition. Using neural networks on music composition is inherently hard because their behavior is not easy to predict nor control. However, the connectionist approach promised to give more flexibility and an ability to create novel situations, which makes it a suitable method for our experiment.

In this paper, a neural network is used to generate melodies, and a set of composition rules (common rules and Bartok's rules) is used as constraints to evaluate them. This way, the results should evolve to satisfy the constraints, and give us Bartok-like melodies with creative variations.

2 Architecture

The most common neural network architecture is the three-layer feed-forward network. However, such a network does not contain a mechanism to remember past history. This makes it unsuitable for the music generation task because repetitive rhythmic patterns and pitches are important ele-

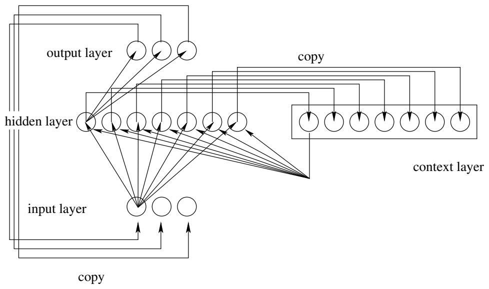


Figure 1: The melody generation network. The values of the output nodes at time t are copied to the input nodes at time $T+1$, and a copy of the hidden layer is saved in the context layer. This way, the network can generate output sequences from an initial starting point. The network is fully connected in the forward direction, and its forward weights are evolved.

ments in composition. Instead, a recurrent network such as the SRN [6] is necessary. Our model is based on the SRN with the following input/output scheme (figure 1): The input layer represents a measure at time T, and the output layer represents the measure at time $T+1$. In other words, we feed the SRN network the current measure as input to get the next measure and this way compose the melody one measure at the time.

2.1 Representing Musical Notation

It is difficult to find a suitable encoding scheme for music because of its many dimensions of variation. Therefore, in this initial stage of development, we limit the available rhythmic notations to five types: a whole note, a half note, a quarter note, a eighth note and a sixteenth note. No rests or dotted notes are used. The range of pitches is also shortened to three octaves, from C2 to C5. The details of the encoding scheme are described below.

2.1.1 Relative Pitch Encoding. Two pitch encoding methods, absolute and relative pitch are possible. Initial experiments showed that relative pitch encoding was more effective at generating good musical sequences. This is an intuitive result based on human performance. People with absolute pitch can identify the pitch without any previous tonal context available. Such people are extremely rare, about one percent of a population of trained music professionals. On the other hand, relative pitch is more of a skill than a talent. People can be trained to excel in pitch identification if known pitches are given beforehand as references [9]. Therefore, it is natural to use relative pitch on a neural network that is trained to produce music based on its past experience.

On the output layer, we use an array of nodes to represent interval steps relative to a reference note.Each node is separated by one half step, the smallest unit in tone. The leftmost node has the most negative number, and the rightmost node has the most positive number. The middle node represents zero difference in pitch.

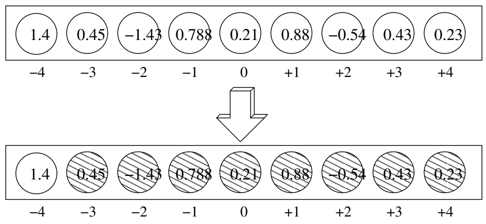


Figure 2: Relative pitch representation. Each output node corresponds to an increase or decrease in pitch of a half-step. The node that has the highest value wins. In this case, the winner is -4. If the reference note is G4, then the result is G4 -4 = D4#.

In some cases, the output will drive the pitch out of the available range of pitches in the representation. For example, if the output from the network continues to raise the pitch higher than the previous pitch, eventually the output pitch will be higher than C5, the upper bound in our experiments. The solution is to lower this pitch by an octave to get perceptually the closest pitch. We apply the same technique to the out of lower bound situation: the pitch is raised by an octave.

2.1.2 Duration Representation. An array of five nodes are used to represent the tone duration. The implementation is similar to the pitch representation, however, each node does not correspond to a specific offset but rather to a duration. The node with the highest value wins and its duration is assigned to the current note. If more than one node end up with the same highest value, the node that corresponds to the longest duration wins.

2.2 Representing Measures

Measures are the building blocks in a melody. Each measure corresponds to the same length of time. The duration of all notes inside a measure must add up to that length. In our experiment we use one unit of time as the length of the measure so that we can fit exactly one whole note, or two half notes, or four quarter notes, or any combinations of notes that sum up to one in a measure. We will call the concatenation of one duration representation and one pitch representation a *D-P* pair. A measure representation consists of sixteen *D-P* pairs (figure 3). A measure is extracted from this representation according to the following algorithm:

One *D-P* pair is decoded at a time starting from the leftmost pair on the output layer. The duration of the note is summed to a variable T. If the duration in T has not exceed one unit

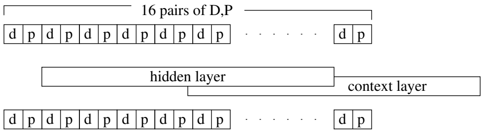


Figure 3: Duration and pitch pairing. One duration representation and one pitch representation are concatenated to form a D-P pair. There are 16 D-P pairs in a measure; only the duration up to one unit of time is used.

of time, we decode next *D-P* pair. If T exceeds one unit of time, we take only a portion of the duration to make T exactly one unit of time.

2.3 Genetic Algorithm

Genetic algorithms (GA) are well suited for music composition because they can explore the solution space in parallel and search for optimal combinations. We use genetic algorithms to evolve a network to produce better melodies. Recently, Moriarty and Miikkulainen [10] developed an efficient neuro-evolution system called SANE (Symbiotic, Adaptive Neuro-Evolution). SANE is quick and explorative in evolving neural networks. Our system uses SANE to effectively guide the evolution to good solutions.

The quality of the melodies produced by the networks is measured based on how well they match a given set of constraints. This general approach has been shown effective in the task of evolving syllable systems [12]. Like that of a syllable system, the quality of a melody cannot be measured quantitatively as a whole. Instead, we evaluate the melody based on how well it satisfies a set of composition rules from music theory. We associate each rule with a weight, and the melody gets a percentage of each weight. Collectively all scores add up to a total that represents the quality of the melody.

There are two constraint categories: rules describing the style of a composer and general rules drawn from music theory. The first three rules below concern specific musical sequences identified as common to Bela Bartok. The fourth rule defines a general constraint on pitch class, and the last three rules represent general composition principles.

2.3.1 X-Cell, Y-Cell, and Z-Cell Structure Constraints.

All notes in the measures are played in a continuous fashion; therefore, when analyzing cell structures, it is convenient to eliminate the measure boundaries and treat the whole melody as one segment. We calculate the pitch difference between successive notes in *half steps*, the unit distance in tonal analysis. Starting from the beginning of the melody, we search for occurrences of X-Cell, Y-Cell and Z-Cell structures. An X-Cell is a group of four notes each separated by a half step in either direction from the previous one; a Y-Cell is a similar group of four notes each separated by two half steps, and a Z-Cell is a group of four notes separated first by two half steps, then by four half steps, and finally by two half steps. The pitches of the notes in a cell may vary as long as their separation matches the cell structure. An example of the X-Cell is C-C#-C-B, where each note is a half step apart. An example of the Y-Cell is C-D-E-F# and of the Z-Cell is G-A-F-G.

To score a melody according to how well it matches a cell structure constraint, we count how many times the cell pattern occurs over the maximum number of patterns, and multiply by the weight:

$$F_s = \frac{N_s}{N_N/4} W_s,$$

where F_s is the fitness function for the constraint (X, Y, or Z-cell), N_s is the number of sequences that contain this cell structure, N_N is the number of notes in the melody, and W_s is the weight of the constraint. Since the cell structures consist of four notes, the maximum number of cells that can occur in the melody is $N_N/4$.

2.3.2 Pentatonic Pitch Class Constraint. A pitch class defines a subset of pitches that can be selected from an octave in the composition. Pitch classes are well defined in music theory and are the basis for euphonious music. *Pentatonic* pitch class, for example, consists of C, D, E, G, and A tones. There are other pitch classes such as Octatonic, Whole-tone, Mixolydian, and Dorian; they are not implemented in the current system. The pitch class constraint is calculated by counting the number of notes that belong to the pentatonic pitch class and divide by the total number of notes, multiplied by the weight:

$$F_c = \frac{N_c}{N_N} W_c,$$

where F_c is the fitness function for this constraint, N_c is the number of notes that are in the pitch class, N_N is the total number of notes in the melody, and W_c is the weight of the pitch class constraint.

2.3.3 Pitch Diversity Constraint. Pitch diversity is a harder constraint to be implement because there is no obvious approach to judge diversity. In principle it would be necessary to look at the entire melody, which would be very time-consuming. Instead, we take a simpler approach: we count the number of measures that have unique pitch sequences. However, if two patterns are the same under transposition[1], they sound alike and should not be counted

[1]Transposition is a technique to produce the same sequence of notes from a different starting pitch. For example, A4-C4-D4 under transposition with E4 becomes E4-F4#-F4. The most common transposition is to raise or lower the segment by an octave.

as different measures. Therefore, we calculate the intervals between pitches and form a signature string unique to that measure and the same measure under all transpositions. For example, the measure C3-D3-D3-B2 has a signature “2u03d”, which stands for two up, zero, and three down. The sum of all unique signatures is divided by the total number of measures and multiplied by its fitness score:

$$F_p = \frac{N_p}{N_M} W_p,$$

where F_p is the fitness function for this constraint, N_p is the number of unique signatures derived from measures, N_M is the number of measures, and W_p is the weight of the pitch diversity constraint.

2.3.4 Rhythmic Diversity Constraint. Rhythmic analysis is easier than pitch analysis because there is no transposition. As with pitch, we use measures as the basis for calculation, and follow the same steps as in pitch diversity analysis. We form a signature string for every measure based on its rhythmic notation. For example a sequence of an eighth note, an eighth note, and a quarter note would have a signature of “884”. The equation is analogous to that of pitch diversity constraint:

$$F_r = \frac{N_r}{N_M} W_r,$$

where F_r is the fitness function for this constraint, N_r is the number of unique signatures derived from measures, N_r, M is the number of measures, and W_r is the weight of the rhythmic diversity constraint.

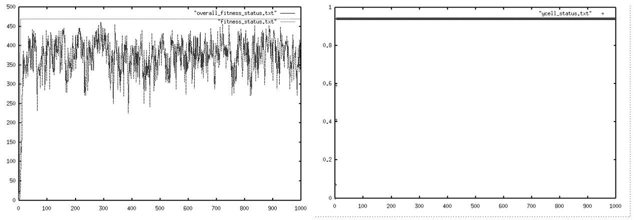
2.3.5 Measure Density Constraint. The Measure Density constraint is included to encourage evolution to favor measures with many notes. This constraint is implemented as follows: calculate the number of notes in the melody and divide by the maximum number of notes that can fit in a melody:

$$F_m = \frac{N_m}{16 N_M} W_m$$

where F_m is the fitness function of this constraint, N_m is the number of notes, N_M is the number of measures in the melody, and W_m is the weight of this constraint. Each measure may have at most 16 sixteenth notes, so the maximum number of notes is $16N_M$.

3 Experiments

The output melodies are shaped by the constraints. If any of the cell constraints is active (i.e. has a large weight), we should see cell structures of that kind in the melody. If the pitch class constraint is on, we expect to hear pitches mostly



(a) average and best fitness (b) y-cell structure

Figure 4: Evolution under the Y-Cell constraint only. The average fitness of the population and the fitness of the best individual over time is plotted on the left. On the right, the percentage of Y-Cell structure is plotted. The graphs show that evolution quickly converges to optimize the constraint.

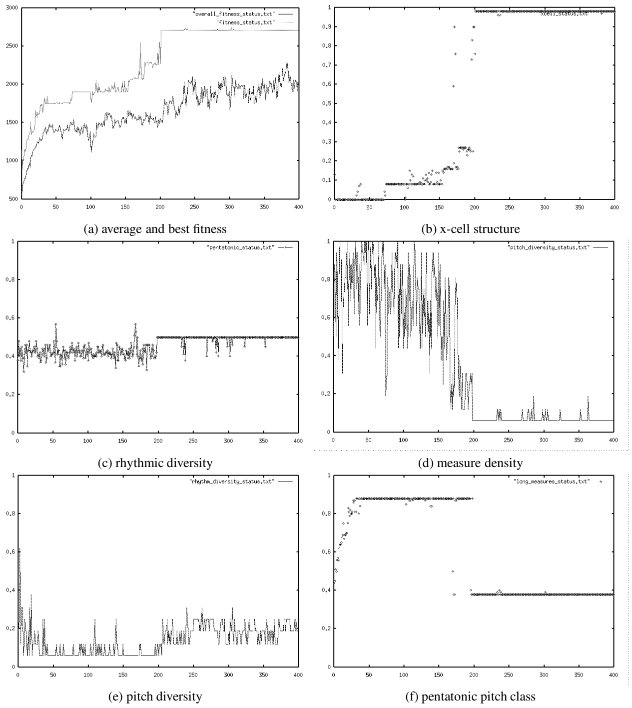
in that pitch class. The diversity constraints should promote variations between measures, and the measure density constraint is designed to control the average number of notes in a measure.

We will test the system in two stages. First, we will disable all constraints but one and evolve the networks using that one constraint. We expect that the resulting melodies express the constraint. After all of the constraints are verified to work alone, we will put them together and observe the different ways in which evolution tries to satisfy them simultaneously.

In the first experiment only the Y-Cell contraint was activated. After 10 generations in the evolution, 95% of the cell patterns were Y-Cells (figure 4). We observed a similar trend when running the system with the X-Cell constraint alone. With the Z-Cell constraint, the system took twice as many generations but did eventually arrive at similar results. Evolution with the Measure Density constraint very quickly favored those networks that produced melodies with lots of short notes in measures. Both diversity constraints worked well too in that there were no similar measures in the resulting melodies. Finally, evolution with the Pentatonic Pitch Class constraint quickly favored those melodies with mostly pentatonic notes.

The above results show that the evolution works well under a simple constraint environment. In the second experiment, the system was tested with several simultaneously active constraints. We enabled all constraints except the Y-Cell constraint and the Z-Cell constraint. Each active constraint was assigned with an equal weight, except the X-Cell constraint, which had a higher weight. We expected the evolution to output melodies with many X-Cells as well as maintaining diversity and density. After several hundred generations in the evolution, the system indeed learned to output this kind of melodies (figure 5).

To illustrate, let us examine two melodies: the first one,



(a) average and best fitness

(b) x-cell structure

(c) rhythmic diversity

(d) measure density

(e) pitch diversity

(f) pentatonic pitch class

Figure 5: Evolution with multiple constraints. The average fitness of the population and the fitness of the best individual over time is plotted on the upper left. The other panels show the extent to which the X-cell structure, pentatonic pitch class, pitch diversity, rhythmic diversity and measure density constraints were satisfied. Different runs find in different tradeoffs between the constraints, resulting in systematically different melodies.

shown in figure 6, was taken at the 194th generation, and the second (in figure 7) was the final best melody at the 200th generation.[2] The statistics describing these melodies are shown in table 1. It is intriguing to compare the styles of these samples. The two melodies scored roughly the same fitness score, but they sound very different. The first melody consist of more dense measures while the other has a lot more X-Cells. Also, the first one had greater pitch diversity but had fewer notes in the pentatonic pitch class. Such a drastic change in style is common in evolution: evolution explores different aspects of style in parallel while continuing to gain higher fitness scores overall. This way, interesting and creative solutions can be generated to the problem of satisfying the musical grammar constraints. Second, there is a remarkable consistency of style within each melody and variation of the basic themes. Such property makes the melodies sound interesting and meaningful, as opposed to mechanical or random.

The consistency with variation property is due to the net-

[2]Sound for these and other example melodies can be found at www.cs.utexas.edu/users/nn/pages/research/neuroevolution.html.

Table 1: Statistics of sample melodies at the 194th and the 200th generation.

Gener-ation	X-Cell	Pitch Class	Pitch Diversity	Rhythmic Diversity	Measure Density
194	25%	39%	25%	6%	88%
200	76%	50%	6%	6%	38%

work architecture used. With SRN, the system learns to output patterns that it has previously generated. As seen in figure 6 and 7, the system naturally adopted transposition as a means to reproduce similar patterns but beginning at a different pitch. Once the system finds a good measure that scores high on all constraints, it uses transposition to fill the melody with the same measures. We also observed that there were slight variations in the measures (figure 6). By altering a few notes in each measure, the melody partly satisfied the pitch diversity constraint and scored 25% of its weight.

4 Discussion and Future Work

The system worked well in several aspects. First, evolution was able to respond to different constraint weights and produce melodies of the desired kind in each case. Second, it generated variation within these constraints, i.e. distinctively different ways to satisfy them. Third, within each melody there was systematic structure, but also a variation of this structure, which gives the melodies a more interesting feel than is common in mechanical composition. These properties result directly from the architecture of the system, that is, from evolution of simple recurrent networks.

However, the system in its present form is limited in many ways. We omitted dotted notes and rests as well as triad and other chords. There is currently no representation for multiple instruments or complements. The rule set in general is rather simple and could be expanded a lot. For example, in the 16th century there was an attempt to formulate melodies into a set of rules [14]. The rules were well documented, and music scholars over the centuries used them for their composition. It would be interesting to include these rules in the system and compare to the music composed at that time.

The most serious challenge, however, is global structure. When listening to the samples, we noticed that the melodies lacked a certain global flow. Melodies were generated one measure at a time. Individually the measures sounded good, and they were diverse, but their overall organization was not satisfactory. It would be possible to add a constraint to the system to define the flow between measures. The challenge

Figure 6: Sample melody at the 194th generation. The measures are dense but only 25% of the notes are members of X-Cells. Most of the measures are transposed from one basic style with some variations.

Figure 7: Sample melody at the 200th generation. The measures are less dense and 75% of the notes are members of X-Cells. All measures are transposed from one basic style with little variation. Half of the notes are in the pentatonic pitch class.

is to describe computationally what a good flow is.

5 Conclusions

The neuroevolution system developed in this paper serves as a basic framework for artificial composition using rules and principles from music theory. Combining common composition rules and specific style descriptions of Bartok, the system generates melodies that locally resemble Bartok's music. Though the results are still simple, they are promising and likely to improve if more composition rules are added to the system.

6 Acknowledgments

This research was funded in part by NFS under Grant #IIS-0083776. Special thanks to Professor Russell Pinkston in the Department of Music, University of Texas at Austin for his assistance in general music theory.

References

[1] Ames C. and Domino, M. (1994). Cybernetic Composer: An Overview. Technical Report, Kurzweil Foundation, Automated Composition Project.

[2] Antokoletz, E. (1981). The Musical Language of Bartok's 14 Bagatelles for Piano. *Tempo*, Vol. 137.

[3] Antokoletz, E. (1984). *The Music of Bela Bartok: A Study of Tonality and Progression in Twentieth Century Music.* Berkeley and Los Angeles: University of California Press.

[4] Bernstein, L. (1976). *The Unanswered Question: Six Talks at Harvard.* Cambridge: Harvard University Press.

[5] Desain, P. and Honing, H. (1992). A connectionist and a traditional AI quantizer, symbolic versus subsymbolic models of rhythm perception. *Music, Mind and Machine: Studies in Computer Music, Music Cognition and Artificial Intelligence, 81–98.* Amsterdam: Thesis Publishers.

[6] Elman, Jeffrey L. (1990). Finding Structure in Time. *Cognitive Science* 14:179–211.

[7] Lerdahl, F. and Jackendoff, R. (1983). *A Generative Theory of Tonal Music.* Cambridge, MA: MIT Press.

[8] Longuet-Higgins, H. C. (1976). Perception of melodies. *Nature* 263:646–653.

[9] Miyazaki K. (1992). Perception of Musical Intervals by Absolute Pitch Possessors. *Music Perception* 9:413–426.

[10] Moriarty, D. E. and Miikkulainen, R. (1996). Efficient reinforcement learning through symbiotic evolution. *Machine Learning* 22:11–32.

[11] Mozer, M. C. (1994). Neural Network Music Composition by Prediction: Exploring the Benefits of Psychoacoustic Constraints and Multi-scale Processing. *Connection Science* 6:247–80.

[12] Redford, M. A., Chen, C-C. J. and Miikkulainen, R. (1998). Modeling the Emergence of Syllable Systems. *Proceedings of the 20th annual Conference of the Cognitive Science Society.* Hillsdale, NJ: Erlbaum.

[13] Riecken R. D. (1994). WOLFGANG—A System Using Emoting Potentials to Manage Musical Design. Technical Report, AT&T Bell Laboratories.

[14] Thakar, M. (1990). Counterpoint study for both composers and performers. *Counterpoint: Fundamentals of Music Making*. New Haven, CT: Yale University Press.

[15] Todd, P. M. and Loy, D. G. (1991). *Music and Connectionism*, ix–xi. Cambridge, MA: MIT Press.