

A Sequential Covering Evolutionary Algorithm for Expressive Music Performance*

Rafael Ramirez and Amaury Hazan and Jordi Mariner and Esteban Maestre

Music Technology Group
Pompeu Fabra University
Ocata 1, 08003 Barcelona, Spain
Tel:+34 935422165, Fax:+34 935422202
{rafael, ahazan, jmariner, emaestre}@iua.upf.es

Abstract

In this paper, we describe an evolutionary approach to one of the most challenging problems in computer music: modeling the knowledge applied by a musician when performing a score of a piece in order to produce an expressive performance of the piece. We extract a set of acoustic features from Jazz recordings thereby providing a symbolic representation of the musician's expressive performance. By applying a sequential covering evolutionary algorithm to the symbolic representation, we obtain an expressive performance computational model capable of endowing a computer generated music performance with the timing and energy expressiveness that characterizes human generated music.

Introduction

Expressive performance is an important issue in music which in the past has been studied using different approaches (e.g. (Gabrielsson, 1999)). The main approaches to empirically study expressive performance have been based on statistical analysis (e.g. (Repp, 1992)), mathematical modelling (e.g. (Todd, 1992)), and analysis-by-synthesis (e.g. (Friberg, 1995)). In all these approaches, it is a person who is responsible for devising a theory or mathematical model which captures different aspects of musical expressive performance. The theory or model is later tested on real performance data in order to determine its accuracy.

In this paper we describe an approach to investigate musical expressive performance based on evolutionary computation. Instead of manually modelling expressive performance and testing the model on real musical data, we let a computer use a sequential covering genetic algorithm to automatically discover regularities and performance principles from real performance data: audio recordings of Jazz standards. The algorithm incrementally constructs a set of rules by learning new rules one at a time, removing the positive examples covered by the latest rule before attempting to learn the next rule. The algorithm provides an interpretable specification

*This work is supported by the Spanish TIC project ProMusic (TIC 2003-07776-C02-01). We would like to thank Emilia Gomez and Maarten Grachten for processing the data, as well as the anonymous reviewers for their insightful comments.
Copyright © 2006, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

of the expressive principles applied to a music interpretation and, at the same time, it provides a generative model of expressive performance, i.e. a model capable of endowing a computer generated music performance with the timing and energy expressiveness that characterizes human generated music

The rest of the paper is organized as follows: Section 2 describes how we extract a set of acoustic features from the audio recordings in order to obtain a symbolic description of the different expressive parameters embedded in the recordings. In Section 3, we describe our evolutionary approach for inducing an expressive music performance computational model. Section 4 reports on related work, and finally Section 5 presents some conclusions and indicates some areas of future research.

Melodic Description

In this section, we summarize how we extract a symbolic description from the monophonic recordings of performances of Jazz standards. We need this symbolic representation in order to apply a sequential covering genetic algorithm to the data. In this paper, our interest is to model note-level transformations such as onset deviations, duration transformations and energy variations. Thus, descriptors providing note-level information are of particular interest.

Algorithms for feature extraction

First of all, we perform a spectral analysis of a portion of sound, called analysis frame, whose size is a parameter of the algorithm. This spectral analysis lies in multiplying the audio frame with an appropriate analysis window and performing a Discrete Fourier Transform (DFT) to obtain its spectrum. In this case, we use a frame width of 46 ms, an overlap factor of 50%, and a Keiser-Bessel 25dB window. Then, we perform a note segmentation using low-level descriptor values. Once the note boundaries are known, the note descriptors are computed from the low-level and the fundamental frequency values.

Low-level descriptors computation

The main low-level descriptors used to characterize expressive performance are instantaneous energy and fundamental frequency.

Energy computation. The energy descriptor is computed on the spectral domain, using the values of the amplitude spectrum at each analysis frame. In addition, energy is computed in different frequency bands as defined in (Klapuri, 1999), and these values are used by the algorithm for note segmentation.

Fundamental frequency estimation. For the estimation of the instantaneous fundamental frequency we use a harmonic matching model derived from the Two-Way Mismatch procedure (TWM) (Maher, 1994). For each fundamental frequency candidate, mismatches between the harmonics generated and the measured partials frequencies are averaged over a fixed subset of the available partials. A weighting scheme is used to make the procedure robust to the presence of noise or absence of certain partials in the spectral data. The solution presented in (Maher, 1994) employs two mismatch error calculations. The first one is based on the frequency difference between each partial in the measured sequence and its nearest neighbor in the predicted sequence. The second is based on the mismatch between each harmonic in the predicted sequence and its nearest partial neighbor in the measured sequence. This two-way mismatch helps to avoid octave errors by applying a penalty for partials that are present in the measured data but are not predicted, and also for partials whose presence is predicted but which do not actually appear in the measured sequence. The TWM mismatch procedure has also the benefit that the effect of any spurious components or partial missing from the measurement can be counteracted by the presence of uncorrupted partials in the same frame.

First, we perform a spectral analysis of all the windowed frames, as explained above. Secondly, the prominent spectral peaks of the spectrum are detected from the spectrum magnitude. These spectral peaks of the spectrum are defined as the local maxima of the spectrum which magnitude is greater than a threshold. The spectral peaks are compared to a harmonic series and a two-way mismatch (TWM) error is computed for each fundamental frequency candidates. The candidate with the minimum error is chosen to be the fundamental frequency estimate.

After a first test of this implementation, some improvements to the original algorithm were implemented to deal with some errors of the algorithm:

- Peak selection: a peak selection routine has been added in order to eliminate spectral peaks corresponding to noise. The peak selection is done according to a masking threshold around each of the maximum magnitude peaks. The form of the masking threshold depends on the peak amplitude, and uses three different slopes depending on the frequency distance to the peak frequency.
- Context awareness: we take into account previous values of the fundamental frequency estimation and instrument dependencies to obtain a more adapted result.
- Noise gate: a noise gate based on some low-level signal descriptor is applied to detect silences, so that the estimation is only performed in non-silent segments of the sound.

Note segmentation

Note segmentation is performed using a set of frame descriptors, which are energy computation in different frequency bands and fundamental frequency. Energy onsets are first detected following a band-wise algorithm that uses some psycho-acoustical knowledge (Klapuri, 1999). In a second step, fundamental frequency transitions are also detected. Finally, both results are merged to find the note boundaries (onset and offset information).

Note descriptor computation

We compute note descriptors using the note boundaries and the low-level descriptors values. The low-level descriptors associated to a note segment are computed by averaging the frame values within this note segment. Pitch histograms have been used to compute the pitch note and the fundamental frequency that represents each note segment, as found in (McNab, 1996). This is done to avoid taking into account mistaken frames in the fundamental frequency mean computation. incrementally constructs a set of rules by learning new rules one at a time, removing the positive examples covered by the latest rule before attempting to learn the next rule. Rules are learned using a genetic algorithm

Learning the expressive performance model

In this section, we describe our inductive approach for learning an expressive music performance model from performances of Jazz standards. Our aim is to obtain a model capable of endowing a computer generated music performance with the expressiveness that characterizes human generated music. That is to say, we intend to generate automatically human-like expressive performances of a piece given an in-expressive description of the piece (e.g. a textual description of its score).

We are aware of the fact that not all the expressive transformations performed by a musician can be predicted at a local note level. Musicians perform music considering a number of abstract structures (e.g. musical phrases) which makes of expressive performance a multi-level phenomenon. In this context, our aim is to obtain a computational model of expressive performance which combines note-level and structure-level information. As a first step in this direction, we have based our musical analysis on the implication/realization model, proposed by Narmour (Narmour, 1990). The Implication/Realization model is a theory of perception and cognition of melodies. The theory states that a melodic musical line continuously causes listeners to generate expectations of how the melody should continue. Any two consecutively perceived notes constitute a melodic interval and if this interval is not conceived as complete, it is an *implicative interval*, i.e. an interval that implies a subsequent interval with certain characteristics. That is to say, some notes are more likely than others to follow the implicative interval. Two main principles recognized by Narmour concern *registral direction* and *intervallic difference*. The principle of registral direction states that small intervals imply an interval in the same registral direction (a small

upward interval implies another upward interval and analogously for downward intervals), and large intervals imply a change in registral direction (a large upward interval implies a downward interval and analogously for downward intervals).

Training data

The training data used in our experimental investigations are monophonic recordings of four Jazz standards (Body and Soul, Once I Loved, Like Someone in Love and Up Jumped Spring) performed by a professional musician at 11 different tempos around the nominal tempo. For each piece, the nominal tempo was determined by the musician as the most natural and comfortable tempo to interpret the piece. Also for each piece, the musician identified the fastest and slowest tempos at which a piece could be reasonably interpreted. Interpretations were recorded at regular intervals around the nominal tempo (5 faster and 5 slower) within the fastest-slowest tempo limits. The data set is composed of 4360 performed notes. Each note in the training data is annotated with its corresponding performed characteristics and a number of attributes representing both properties of the note itself and some aspects of the context in which the note appears. Information about the note include note duration and the note metrical position within a bar, while information about its melodic context include performed tempo, information on neighboring notes as well as the Narmour group in which the note appears in third position. Intuitively, the Narmour group in which the note appears in third position provides an indicator of how expected the note is.

Learning task

In this paper, we are concerned with note-level expressive transformations, in particular transformations of note duration, onset and energy. Initially, for each expressive transformation, we approach the problem as a classification problem, e.g. for note duration transformation we classify each note to belong to one of the classes *lengthen*, *shorten* or *same*. Once we obtain a classification mechanism capable of classifying all notes in our training data, we apply a regression algorithm in order to produce a numerical value representing the amount of transformation to be applied to a particular note. The complete algorithm is detailed in the next section.

The performance classes that interest us are *lengthen*, *shorten* and *same* for duration transformation, *advance*, *delay* and *same* for onset deviation, and *soft*, *loud* and *same* for energy variation. A note is considered to belong to class *lengthen*, if its performed duration is 20% longer (or more) than its nominal duration, e.g. its duration according to the score. Class *shorten* is defined analogously. A note is considered to be in class *advance* if its performed onset is 5% of a bar earlier (or more) than its nominal onset. Class *delay* is defined analogously. A note is considered to be in class *loud* if it is played louder than its predecessor and louder than the average level of the piece. Class *soft* is defined analogously. We decided to set these boundaries after experimenting with different ratios. The main idea was to guarantee that a note

classified, for instance, as *lengthen* was purposely lengthened by the performer and not the result of a performance inexactitude.

Algorithm

We applied a genetic sequential covering algorithm to the training data. Roughly, the algorithm incrementally constructs a set of rules by learning new rules one at a time, removing the positive examples covered by the latest rule before attempting to learn the next rule. Rules are learned using a genetic algorithm with the usual parameters r , m and p respectively determining the fraction of the parent population replaced by crossover, the mutation rate, and population size. We set these parameters as follows: $r = 0.8$, $m = 0.05$ and $p = 200$. For each class of interest (e.g. *lengthen*, *shorten*, *same*), we collect the rules with best fitness during the evolution of the population. For obtaining rules for a particular class of interest (e.g. *lengthen*) we consider s negative examples the examples of the other two complementary classes (e.g. *shorten* and *same*). It is worth mentioning that although the test was running over 40 generations, the fittest rules were obtained around the 20th generation. Once we obtain the set of rules covering all the training examples, for each rule, we apply linear regression to the examples covered by the rule in order to obtain a linear equation predicting a numerical value. This leads to a set of rules producing a numerical prediction and not just a nominal class prediction. The algorithm is as follows:

```
GeneticSeqCovAlg(Class, Fitness, Threshold,
                 p, r, m, Examples)
Pos = examples which belong to Class
Neg = examples which do not belong to Class
Learned_rules = {}
While Pos do
  P = generate p hypothesis at random
  for each h in P, compute fitness(h)
  while max(fitness(h)) < Threshold do
    create a new generation Pnew
    P = Pnew
    for each h in P, compute fitness(h)
  Return the hypothesis Newrule from P
    that has the highest fitness
  Rpos = members of Pos covered by NewRule
  compute PredictedValue(Rpos)
  NumericNewRule = NewRule with Class
    replaced by Regression(Rpos)
  Learned_rules = Learned_rules
    + NumericNewrule
  Pos = Pos - Rpos
Return Learned_rules
```

The outer loop learns new rules one at a time, removing the positive examples covered by the latest rule before attempting to learn the next rule. The inner loop performs a genetic search through the space of possible rules in search of a rule with high accuracy. At each iteration, the outer loop adds a new rule to its disjunctive hypothesis, *Learned_rules*. The effect of each new rule is to generalize the current disjunctive hypothesis (i.e. increasing the number of instances it classifies as positive) by adding a new disjunct. At this level, the search is a

specific-to-general search starting with the most specific hypothesis (i.e. the empty disjunction) and terminating when the hypothesis is sufficiently general to cover all training examples. `NumericNewRule` is a rule where the consequent `Regression(Rpos)` is a linear equation $X = w_0 + w_1 * a_1 + w_2 * a_2 + \dots + w_k * a_k$ where X is the predicted value expressed as a linear combination of the attributes a_1, \dots, a_k of the training examples with predetermined weights w_0, \dots, w_k . The weights are calculated using the set of positive examples covered by the rule `RPOS` by linear regression. The inner loop performs a fine grained search to determine the exact form of each new rule. In the inner loop, a new generation is created as follows:

- **Select:** probabilistically select $(1 - r)p$ members of `P` to add to `PS`. The probability of $Pr(h_i)$ of selecting hypothesis h_i from `P` is

$$Pr(h_i) = \frac{Fitness(h_i)}{\sum(h_j)} \quad (1 \leq j \leq p)$$

- **Crossover:** probabilistically select $(r * p)/2$ pairs of hypothesis from `P` (according to $Pr(h_i)$ above). For each pair, produce an offspring by applying the crossover operator (see below) and add it to `PS`.
- **Mutate:** Choose m percent of the members of `PS` with uniform probability and apply the mutation operator (see below).

Hypothesis representation. The hypothesis space of rule preconditions consists of a conjunction of a fixed set of attributes. Each rule is represented as a bit-string as follows: the previous and next note duration are represented each by five bits (i.e. much shorter, shorter, same, longer and much longer), previous and next note pitch are represented each by five bits (i.e. much lower, lower, same, higher and much higher), metrical strength by five beats (i.e. very weak, weak, medium, strong and very strong), tempo by three bits (i.e. slow, nominal and fast) and Narmour group by three bits. For example in our representation the rule

“if the previous note duration is much longer and its pitch is the same and it is in a very strong metrical position and the current note appears in Narmour group R then lengthen the duration of the current note”

is coded as the binary string:

00001 11111 00100 11111 00001 111 110 001

The exact meaning of the adjectives which the particular bits represent are as follows: previous and next note durations are considered much shorter if the duration is less than half of the current note, shorter if it is shorter than the current note but longer than its half, and same if the duration is the same as the current note. Much longer and longer are defined analogously. Previous and next note pitches are considered much lower if the pitch is lower by a minor third or more, lower if the pitch is within a minor third, and same if

it has same pitch. Higher and much higher are defined analogously. The note’s metrical position is very strong, strong, medium, weak, and very weak if it is on the first beat of the bar, on the third beat of the bar, on the second or fourth beat, offbeat, and in none of the previous, respectively. The piece was played at slow, nominal, and fast tempos if it was performed at a speed slower of more than 15% of the nominal tempo (i.e. the tempo identified as the most natural by the performer), within 15% of the nominal tempo, and faster than 15% of the nominal tempo, respectively. In the case of the note’s Narmour groups we decided to code only one Narmour group for each note. This is, instead of specifying all the possible Narmour groups for a note, we select the one in which the note appears in third position (if there is no such group, we consider one in which the note appears either in first or second position, in that order).

Genetic operators. We use the standard single-point crossover and mutation operators with two restrictions. In order to perform a crossover operation of two parents the crossover points are chosen at random as long as they are on the attributes sub string boundaries. Similarly the mutation points are chosen randomly as long as they do not generate inconsistent rule strings, e.g. only one class can be predicted so exactly one 1 can appear in the last three bit sub string.

Fitness function. The fitness of each hypothesized rule is based on its classification accuracy over the training data. In particular, the function used to measure fitness is

$$\frac{tp^{1.15}}{(tp+fp)}$$

where tp is the number of true positives and fp is the number of false positives.

Results

It is always difficult to evaluate formally a model which captures subjective knowledge, as it is the case of an expressive music performance model. The ultimate evaluation may consist of listening to the transformations the model performs. However, and despite the relatively small amount of training data, the induced model seems to accurately capture the musician’s expressive performance transformations. The correlation coefficient for the onset, duration and energy sub models is 0.75, 0.84 and 0.86, respectively. These numbers were obtained by performing a 10-fold cross validation on the data. At each fold, we removed the performances similar to the ones selected in the test set, i.e. the performances of the same piece at similar tempos. We ran the sequential covering genetic algorithm 20 times in order to observe the differences between the correlation coefficient of different runs. We observed no substantial differences.

The use of an evolutionary algorithm for inducing an expressive performance model provides the possibility of examining the model as it ‘evolves’. This is, it is possible to examine and interpret the rules induced by the algorithm

at each population generation. In particular, we are interested in interpreting high-accuracy rules in late populations (25th generation or later). The evolutionary approach to expressive performance modeling permits to guide the model search in a natural way. For instance, we have experimented by imposing restrictions on the general shape of rules in order to prioritize simple interpretable rules, e.g. rules with blocks of 1's.

We examined some of the classification rules the algorithm induced (before replacing the class with the numerical predicted value). Some of these rules proved to be of musical interest and correspond to intuitive musical knowledge. In order to illustrate the types of rules found let us consider some examples of duration rules:

RULE1: 11111 01110 11110 00110 00011 010 010 001

“At nominal tempo, if the duration of the next note is similar and the note is in a strong metrical position and the note appears in a D Narmour group then lengthen the current note.”

RULE2: 00111 00111 00011 01101 10101 111 111 100

“If the previous and next notes durations are longer (or equal) than the duration of the current note and the pitch of the previous note is higher then shorten the current note.”

Related work

Evolutionary computation

Evolutionary computation has been considered with growing interest in musical applications. It has often been used for compositional purposes (Horner, 1991), either to generate melodies (Dahlstedt, 1999) or rhythms (Tokui, 2000). In (Phon-Amnuaisuk, 1999) the harmonization subtask of composition is addressed, and a comparison between a rule-based system and a genetic algorithm is presented.

Evolutionary computation has also been considered for improvisation applications such as (Biles, 1994), where a genetic algorithm-based model of a novice Jazz musician learning to improvise was developed. The system evolves a set of melodic ideas that are mapped into notes considering the chord progression being played. The fitness function can be altered by the feedback of the human playing with the system.

Nevertheless, very few works focusing on the use of evolutionary computation for expressive performance analysis have been done. To the best of our knowledge, the only work in this direction is the work by Grachten (Grachten, 2004) where the weights of the edit distance operations are optimized by a genetic algorithm in order to annotate a human Jazz performance.

Other machine learning techniques

Previous research in learning sets of rules in a musical context has included a broad spectrum of music domains. The

most related work to the research presented in this paper is the work by Widmer (Widmer, 2002). Widmer has focused on the task of discovering general rules of expressive classical piano performance from real performance data via inductive machine learning. The performance data used for the study are MIDI recordings of 13 piano sonatas by W.A. Mozart performed by a skilled pianist. In addition to these data, the music score was also coded. The resulting substantial data consists of information about the nominal note onsets, duration, metrical information and annotations. When trained on the data, the inductive rule learning algorithm discovered a small set of 17 quite simple classification rules that predict a large number of the note-level choices of the pianist. Widmer's work differ from ours in that he is interested in global (i.e. piece-level) tempo and energy transformations while our interest is in note-level timing and energy transformations.

Tobudic et al. (Tobudic, 2003) describe a relational instance-based approach to the problem of learning to apply expressive tempo and dynamics variations to a piece of classical piano music, at different levels of the phrase hierarchy. The different phrases of a piece and the relations among them are represented in first-order logic. The description of the musical scores through predicates provides the background knowledge. The training examples are encoded by another predicate whose arguments encode information about the way the phrase was played by the musician. Their learning algorithm recognizes similar phrases from the training set and applies their expressive patterns to a new piece. Our work differs from Tobudic's work in the same way as it differs from that of Widmer since both study classical piano expressive performance.

Lopez de Mantaras et al. (Lopez de Mantaras, 2002) report on a performance system capable of generating expressive solo performances in jazz within a case-based reasoning system. Their system focuses on note onset, duration and energy. However, their system does not allow to examine the way it makes predictions.

Conclusion

This paper describes an evolutionary computation approach for learning an expressive performance model from recordings of Jazz standards by a skilled saxophone player. Our objective has been to find a computational model which predict how a particular note in a particular context should be played (e.g. longer or shorter than its nominal duration). In order to induce the expressive performance model, we have extracted a set of acoustic features from the recordings resulting in a symbolic representation of the performed pieces and then applied a sequential-covering genetic algorithm to the symbolic data and information about the context in which the data appear. Despite the relatively small amount of training data, the induced model seems to accurately capture the musician's expressive performance transformations. In addition, some of the classification rules induced by the algorithm proved to be of musical interest. We plan to increase the amount of training data as well as experiment with different information encoded in it. Increasing

the training data, extending the information in it and combining it with background musical knowledge will certainly generate a more accurate model. Another research line is to extend our model to be able to predict not only note-level timing and energy transformations but also intra-note expressive features such as vibrato, and instantaneous energy. We plan to extend the model by characterizing the notes in the data set by their pitch, energy, and timbre features and learn to predict these features of a note according to its musical context.

References

Biles, J. A. (1994). GenJam: A genetic algorithm for generating Jazz solos. In ICMC Proceedings 1994.

Dahlstedt, P., and Nordahl, M. Living Melodies: Coevolution of Sonic Communication First Iteration Conference on Generative Processes in the Electronic Arts, Melbourne, Australia, December 1-3 1999.

De Jong, K.A. et al. (1993). Using Genetic Algorithms for Concept Learning. *Machine Learning*, 13, 161-188.

Friberg, A. (1995). A Quantitative Rule System for Musical Performance. PhD Thesis, KTH, Sweden.

Gabrielsson, A. (1999). The performance of Music. In D.Deutsch (Ed.), *The Psychology of Music* (2nd ed.) Academic Press.

Grachten, M., Luis Arcos, J., and Lopez de Mantaras, R. (2004). Evolutionary Optimization of Music Performance Annotation.

Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.

Horner, A., and Goldberg, 1991. Genetic Algorithms and Computer-Assisted Music Composition, in proceedings of the 1991 International Computer Music Conference, pp. 479-482.

Klapuri, A. (1999). Sound Onset Detection by Applying Psychoacoustic Knowledge, Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP.

Lopez de Mantaras, R. and Arcos, J.L. (2002). AI and music, from composition to expressive performance, *AI Magazine*, 23-3.

Maher, R.C. and Beauchamp, J.W. (1994). Fundamental frequency estimation of musical signals using a two-way mismatch procedure, *Journal of the Acoustic Society of America*, vol. 95 pp. 2254-2263.

McNab, R.J., Smith L.I. A. and Witten I.H., (1996). Signal Processing for Melody Transcription, SIG working paper, vol. 95-22.

Narmour, E. (1990). *The Analysis and Cognition of Basic Melodic Structures: The Implication Realization Model*.

University of Chicago Press.

Phon-Amnuaisuk, S., and A. Wiggins, G. (1999?) The Four-Part Harmonisation Problem: A comparison between Genetic Algorithms and a Rule-Based System.

Repp, B.H. (1992). Diversity and Commonality in Music Performance: an Analysis of Timing Microstructure in Schumann's 'Traumerei'. *Journal of the Acoustical Society of America* 104.

Tobudic A., Widmer G. (2003). Relational IBL in Music with a New Structural Similarity Measure, Proceedings of the International Conference on Inductive Logic Programming, Springer Verlag.

Todd, N. (1992). The Dynamics of Dynamics: a Model of Musical Expression. *Journal of the Acoustical Society of America* 91.

Tokui, N., and Iba, H. (2000). Music Composition with Interactive Evolutionary Computation.

Widmer, G. (2002). Machine Discoveries: A Few Simple, Robust Local Expression Principles. *Journal of New Music Research* 31(1), 37-50.