Audio Oracle analysis of Musical Information Rate

Shlomo Dubnov
Music Department
University of California in San Diego
Email: sdubnov@ucsd.edu

Gérard Assayag
Ircam - UMR CNRS STMS
Paris, France
Email: assayag@ircam.fr

Arshia Cont
Ircam - UMR CNRS STMS
Paris, France
Email: Arshia.Cont@ircam.fr

***Abstract*—This paper presents a method for analysis of changes in information contents in music based on an audio representation called Audio Oracle (AO). Using compression properties of AO we estimate the amount of information that passes between the past and the present at every instance in a musical signal. This formulation extends the notion of Information Rate (IR) to individual sequences and allows an optimal estimation of the AO threshold parameter. We show that changes in IR correspond to significant musical structures such as sections in a sonata form. Relation to musical perception and applications for composition and improvisation are discussed in the paper.**

I. Introduction

Musical Information Dynamics is a study of changes in information content in music that correlate with musically significant events. Moreover, it is assumed that such structures could be captured by cognitive processes related to music perception dealing with processes of learning and detection of regularities in music according to an idea that listening to music consists of forming expectations and continual fulfillment or denial thereof [1]. Recently several information theoretic measures of audio structure have been proposed in attempt to characterize musical contents according to its predictive structure [2],[3],[4],[5]. Information Rate measures the mutual information between past and present in music, and has been applied both to symbolic (MIDI) representations and audio recordings.

Audio Oracle is an indexing structure for audio data that captures repeating sub-clips of variable length called "audio factors" [6]. This structure extends a string matching algorithm called Factor Oracle (FO) [7] to metric space, allowing detection of approximate repetitions in audio signals. Applications of AO include fast retrieval and creative applications such as improvisation and computer aided composition. A useful property of AO is that it due to common suffix representation, it allows recombination of sub-clips in a manner that assures continuity between splice points. This accomplishes effectively a new method for style imitation [8], texture synthesis [9], and audio mosaicing [10], were new variations on musical material are derived through reshuffling of sub-clips while maintaing an overall resemblance to the original sound source.

A. Information Rate

The notion of IR considers the relative reduction of uncertainty of the present in a signal when the past is known. In its original statistical formulation this reduction was measured in terms of mutual information carried between the past $x_{past} = \{x_1, x_2, ..., x_{n-1}\}$ and the present x_n of a signal. The mutual information between the two variables x_{past} and x_n is

$$I(x_{past}, x_n) \quad = \quad H(x_n) - H(x_n|x_{past}) \qquad (1)$$

where $H(x) = -\Sigma P(x) log_2 P(x)$ is the Entropy of variable x with distribution $P(x)$.

In the current formulation of IR we will abandon the statistical formulation but retain the idea that IR measures the relative reduction in uncertainty or complexity of a signal given information about its past. Accordingly, we define a new deterministic IR as

$$IR(x_{past}, x_n) \quad = \quad C(x_n) - C(x_n|x_{past}) \qquad (2)$$

where $C(\cdot)$ is a compression algorithm or some computational measure of signal complexity measured in terms of the number of bits required to represent the data with or without prior context. In the following we will show how AO can be used to define and compute these measures.

B. Audio Oracle

Audio Oracle accepts continuous (audio) signal stream as input, transforms it into a sequence of feature vectors and submits these vectors to AO analysis. AO outputs an automaton that contains pointers to different locations in the audio data that satisfy certain similarity criteria, as found by the algorithm. The resulting automaton is passed next to oracle compression module that will be described in the next section.

Algorithms 1 and 2 demonstrate psuedo-codes for Audio Oracle construction. During the online construction, the algorithm accepts audio frame descriptors (user-defined audio features) as vectors σ_i for each time-frame i and updates audio oracle in an incremental manner. Algorithm 1 shows the main online audio oracle construction algorithm.

Algorithm 1 On-line construction of Audio Oracle

Require: Audio stream as $S = \sigma_1 \sigma_2 \cdots \sigma_N$
1: Create an oracle P with one single state 0
2: $S_P(0) \leftarrow -1$
3: **for** $i = 0$ to N **do**
4: $Oracle(P = p_1 \cdots p_i) \leftarrow Add - Frame\,(Oracle(P = p_1 \cdots p_{i-1}), \sigma_i)$
5: **end for**
6: **return** Oracle $(P = p_1 \cdots p_N)$

Algorithm 1 calls the function `Add-Frame` described in algorithm 2 which updates the audio oracle structure using the latest received frame descriptions. This function works very similar to Factor Oracle except that (1) it accepts *continuous* data flow rather than symbolic data, (2) does not assign symbols to transitions and instead each state has a one-to-one correspondence with frames in audio buffer, and (3) it uses a distance function along with a threshold θ to assign the degree of similarity between frame descriptions. The set of links in Audio Oracle are forward arrows $\delta(i, \sigma)$ and suffix links $S_p(k)$.

Algorithm 2 `Add-Frame` function: Incremental update of Audio Oracle

Require: Oracle $P = p_1 \cdots p_m$ and Audio Frame descriptor vector σ

Create a new state $m + 1$
Create a new transition from m to $m+1$, $\delta m, \sigma = m+1$
$k \leftarrow S_P(m)$
while $k > -1$ **do**
  Calculate distances between σ and S
  Find indexes of frames in S whose distances from σ are less than θ
  if There are indexes found **then**
    Create a transition from state k to $m + 1$, $\delta(k, \sigma) = m + 1$
    $k \leftarrow S_P(k)$
  end if
end while
if $k = -1$ (no suffix exists) **then**
  $s \leftarrow 0$
else
  $s \leftarrow$ where leads the *best* transition (min. distance) from k
end if
$S_{p\sigma} \leftarrow s$
return Oracle $P = p_1 \cdots p_m \sigma$

Similar to the Factor Oracle algorithm, forward transitions correspond to states that can produce *similar patterns* with alternative continuations by continuing forward, and suffix links that correspond to states that share the *largest similar sub-clip* in their past when going backward.

II. Complexity and IR estimation from AO

In this section we propose a measure of complexity of an audio signal based on a compression method that was developed for the Factor Oracle. It should be noted that we do not consider compression for reducing the size of the audio file, and that the features used in AO are lossy and do not allow recovery of the original signal. In this paper compression is used to prune unnecessary suffixes in AO so that only the longest repeated segments are retained. The resulting structure will be used to define the conditional complexity of frame x_n given its past $C(x_n|x_{past})$. We will use AO also to define the unconditional complexity $C(x_n)$ and then use the difference between the two measures to compute the IR.

In order to use FO for compression, the length of a longest repeated suffix needs to be calculated. This is approximated through a related quantity called LRS that increments recursively the length of the common suffix between two positions in the signal - the immediate past of the current frame and immediate past at a location pointed to by the suffix link. Every time a new frame $(i + 1)$ is added, the suffix link and LRS for this frame are computed. If the resulting suffix link points to state 0 this means that no suffix was found since the distance between the new frame and all previous frames exceeded a threshold θ. In such case this new frame has to be individually encoded. Otherwise if suffix link to a previous location in the sequence is found and the length of the repeating suffix is smaller then the number of steps passed since the last encoding event then the whole preceding segment is encoded as a pair $(length, position)$. The amount of memory required to store such pair is $log_2(M) + log_2(N))$ bits, where N is the range of possible positions limited by the eventual duration of the signal, and $M = max(LRS)$ is the length of the maximal longest repeated suffix. Since both N and $max(LRS)$ are unknown until the sequence is presented in full, we need either to assume some a-priori bounds on these values or wait until the AO analysis is completed.

For example, in the symbolic case the word *aabbabbabbab* will be encoded as follows. The first letter will be encoded using the letter *a* using 1 bit over an alphabet $\{a, b\}$. The next occurrence of *a* can be encoded as a pair $(1, 1)$, but since encoding it will take more bits then encoding *a* individually, we will use the shorter encoding and proceed to encoding next *b* individually and then deciding between representing the following *b* as a pair (1,3) or as a single letter, choosing the later option. The compression advantage appears for the remaining portion of the string. According to the encoding method explain in detail in [11], this part can be encoded as a pair $(8, 2)$, which will take $log_2(8) + log_2(12) = 6.58$ bits, practically offering 1 bit saving compared to encoding of the 8 characters individually. Of course in real cases with a larger alphabet and longer strings the compression advantages becomes more evident and it is comparable and often better then other state of the art lossless compressions methods.

In the AO case, the symbols are replaced by feature vectors and the suffix links are derived through the AO algorithm. In order to define a quantity equivalent to the size of the alphabet in the symbolic case[1] the number of forward links originating from state 0 are considered as the number of distinct elements in AO. Representation of this set requires allocating $log_2|\delta(0, :)|$ bits, where $|\delta(0, :)|$ is the cardinality of the set of forward transitions from state 0. Accordingly we use this number of bits as a measure of unconditional complexity $C(x_n)$, independently of n.

[1]Another way to perform context based compression is by quantizing the audio feature vectors and using the standard compression of FO, but this is exactly the situation that AO tries to avoid.

Definition 1: $C(x_n) = log_2|\delta(0,:)|,\ \forall n$

Let us denote by $K(i)$ the array that contains the states where encoding occurs during the compression pass. An algorithm for computing K is as follows

Algorithm 3 Compression Pass over AO

Require: Array containing the length of repeated suffixes for every state $LRS(i), i = 1 \cdots N$

Create an array K with initialization $K = \{1\}$
for $i = 0$ to $N - 1$ **do**
  if $LRS(i+1) < i - K(end) + 1$ **then**
    $K \leftarrow K \cup \{i\}$
  end if
end for
return Array K

Using K we compute $C(x_n|x_{past})$ over segments of AO as follows. For every state i the number of bits required to represent that state is equal to the number of bits required to represent the pair $(length, position)$ divided by the length of the block. Accordingly, we define the conditional complexity as follows

Definition 2: If state n occurs in a segment between encoding events $K(i)$ and $K(i+1)$, the conditional complexity of an event equals to the average bit per sample of this segment

$$C(x_n|x_{past}) = \frac{log_2(N) + log_2(M)}{K(i+1) - K(i)}, \tag{3}$$

where $M = max(LRS)$, and N is the sequence length.

Combining the two notions of unconditional and conditional complexity, we derive an algorithm for Audio Oracle Information Rate (AO-IR) according to equation (2)

Algorithm 4 Information Rate using AO

Require: Array K containing a list of AO encoding event occurrences, unconditional complexity $C = log_2(|\delta(0,:)|$, $M = max(LRS)$, and sequence length N

for $i = 1$ to $|K| - 1$ **do**
  $L = K(i+1) - K(i)$
  $IR[K(i) : K(i+1)] = max(C - \frac{log_2(N)+log_2(M)}{L}, 0)$
end for
return Array IR

A. Automatic Threshold Selection

In order to perform AO analysis it is important to set correctly the threshold parameter θ in AO. When a threshold is too high many frames will be considered similar and the resulting AO representation will resemble that of a constant sequence, as explained below. If the threshold is too low, many frames will be considered different, resulting in what is effectively an equivalent of a random sequence. Accordingly, it is important to find a threshold value where significant changes in musical information dynamics are detected, and the IR measure can be used for automatic detection of the optimal threshold as follows: We define a total IR measure as the sum of the IR values over all states in the analyzed sequence. It is assumed that a good AO analysis corresponds to a situation where IR is high so that we favor AO solutions that capture most of the mutual information between past and present.

The choice of threshold values has an "inverted U function" behavior. For very small threshold AO will hardly detect repetitions and most of the suffix links will point to state 0, and the conditional complexity will be close to the unconditional one resulting in low IR. In the case of a high threshold many states will resemble each other and suffix links will point to a states in the immediate past. Accordingly LRS will grow almost linearly with sequence length. In such case the compressed encoding will be very efficient since there will be only few distinct frames and most of the segments will be encoded by suffix length-position pairs. The uncompressed complexity in this case will be small as well since only few suffix links will point back to state 0, and by construction of AO there will be only few forward transitions originating from state 0, lowering the count of unconditional complexity. An optimal sequence that has high overall IR will be the one with relatively many forward transitions from state 0, but that can be effectively compressed using suffix links.

Figure 1 shows the results of Total IR for various threshold values. The value with highest total IR was selected as the optimal threshold for our following IR experiments and for the comparison to human analysis as described below.

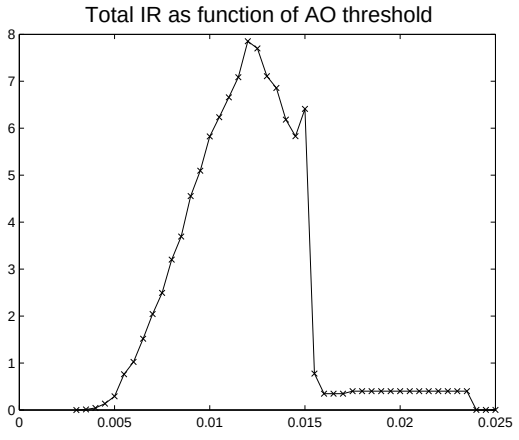

Fig. 1. Sum of AO derived IR values as a function of a threshold θ that was used for construction of the AO. The highest total IR value is used to determine the optimal θ

III. Evaluation

As an evaluation of the AO-IR analysis method we present a detailed analysis of Beethoven Piano Sonata No. 1 in F minor. The recording used in this analysis is by Xenia Knorr, available from Classical Archives[2]. Musicological analysis based on

[2]http://www.classicalarchives.com/artist/4028.html

musical score reveals two repetitions that correspond in the recording to two segments approximately 50 sec. each of an exposition section being played twice (Da Capo), followed by a development section starting at 107 sec., and a recapitulation section starting at 153 sec. The development section appears between 107-153 sec., starting with a repetition of first theme. A change in musical texture occurs at 135 sec. (bar 81) leading to a cadence at 146 sec. (bar 93). A thirty second note idea from the first theme appears in fragments as the tonality modulates back to F minor, with recapitulation occurring at 153 sec (bar 101). Around 160 sec. (bar 108) a silence occurs at a fermata after the first theme reaches a half cadence. This is followed by appearance of the theme in the bass. The closing theme enters at 195 sec. (bar 140), leading to the end of the piece.

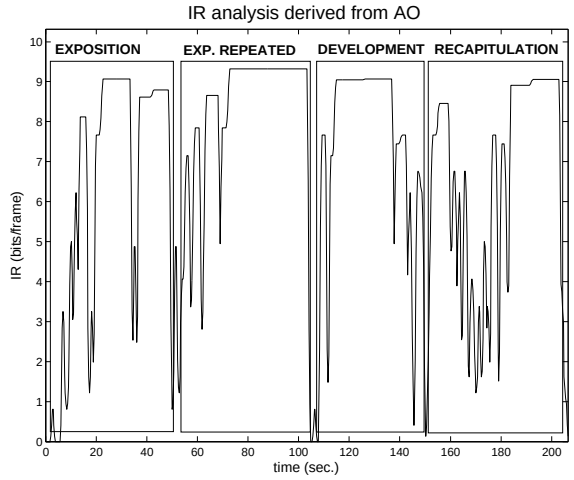

Fig. 2. AO derived IR measure for Beethoven Sonata No. 1

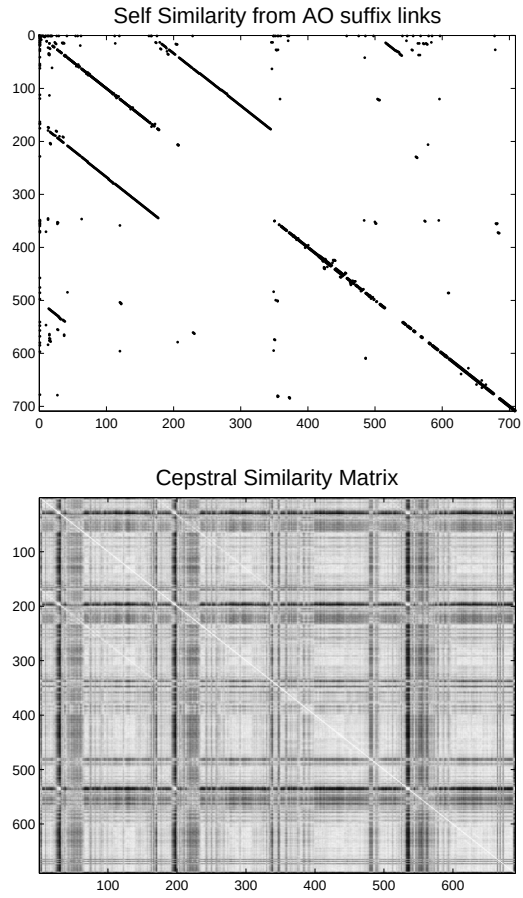

Fig. 3. Self Similarity based on AO Suffix Links (top) and Cepstral features (bottom)

One observes that large drops in AO-IR correspond to significant changes in musical structure. The cepstral features were insufficient to recognize similarity in reappearance of the theme in the bass in the recapitulation section.

A. Comparison to Spectral Clustering and Recurrence analysis

In addition to human expert analysis of the sonata, we compared the AO-IR results to Recurrence Profile derived from Self Similarity matrix of the audio features [12], as explained below. Figure 3 shows the Ceptsral Self Similarity Matrix versus the similarity derived from plotting the suffix links of the AO. Self Similarity matrix is obtained by calculating the pairwise distance between all analysis frames of the entire audio. With proper normalization, this matrix can be interpreted as a transition probability matrix where each entry i, j is considered as probability of transition from frame i to j.

This can be considered as a very crude statistical model of the musical structure that assumes a first order Markov model with transition probability between frames being proportional to similarity between their features (such as cosine distance between spectral vectors). A stationary vector can be derived through eigenvector analysis of of the transition matrix $\mathbf{P}$, finding a vector v so that $v = \mathbf{P}v$. This vector can be interpreted as a profile showing recurrence tendency among the different frames (called Recurrence Profile), with frames belonging to large blocks of similar type of audio tending to have high probability of recurrence.

This analysis method belongs to the broad class of clustering methods known as Spectral Clustering that uses pairwise similarities to segment the data into groups of related elements. The methods differ by the ways they normalize the similarity matrix and the ways they consider the different eigenvectors, followed by a thresholding or grouping step where labeling of the data points is determined according to eigenvector values [13]. The AO-IR analysis is not preforming an actual segmentation, though this is one of its potential applications. Motivated by earlier evidence that Spectral Recurrence is related to human perception of Familiarity [3], we claim that AO-IR would serve even better to capture this aspect of music perception. One notable difference between Spectral Clustering, AO-IR and human reactions is that Spectral Clus-

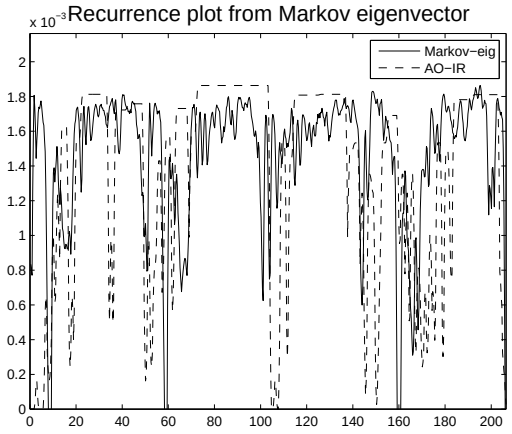

Fig. 4. Recurrence profile using the first eigenvector of a Markov Model derived from Cepstral Self Similarity. This graph is compared to the AO based IR measure (dashed)

tering considers the whole peace at once, thus disregarding the incremental learning or memorization aspects that are done by humans as they listen to the musical work for the first time. The AO based IR measure is incremental as it uses the suffix information from the past but not from the future.

IV. Conclusion

The ability to automatically characterize structural properties of music is important for understanding the relation between musical organization and perception, as well as for applications in computer audition and machine improvisation. Musical Information Dynamics considers the listening act as a process of information processing where anticipations or predictions regarding future musical materials are formed, thus effectively reducing the uncertainty about newly arriving musical data based on its past. It was shown earlier that Recurrence Profile significantly correlates to human judgments of Familiarity and that it is a special case of a more general model-based IR measure of mutual information between past and present in a signal [14]. The current paper presents a significant improvement to the Recurrence Profile analysis method by using information about variable length repetition structures that are detected by AO.

It is also interesting to note that eighty years ago Birkhoff formalized the notion of beauty as the ratio between order and complexity [15]. This measure was later formulated in terms of information theory [16] using the notion of compressed versus uncompressed representations. Complexity is measured in terms of uncompressed data size, or unconditional entropy, while order becomes the difference between the size of unconditional and conditional representations of the data, i.e. when past is used for compression.

IR measure based on AO can be seen as a meta-feature that summarizes repetitions structures found by AO into higher level representations. The low level suffix structure has been used effectively for machine improvisation by creating stylistic re-injections into music performed by a human improvisor. The re-injections, although stylistically coherent, did not take into account larger musical structures. Using the proposed method larger scale planning can be conceived, thus adding to the machine improvisor an ability to track aspects of musical form or control the perception of familiarity in the materials it generates.

Another application of AO is its use for Spine structure for the IEEE 1599 multilayer encoding format [17]. In this format the Structural Layer deals with identifying music objects and their relationships and allows representation of different kinds of musicological analyses through a graphical formalism called Petri Nets. Learning Petri Nets from AO can be suggested as an interesting future direction for research.

References

[1] L. B. Meyer, *Emotion and Meaning in Music*, Chicago: Chicago University Press, 1956.

[2] S. Dubnov. *Spectral anticipations*, Computer Music Journal, 30(2):6383, 2006.

[3] S. Dubnov, S. McAdams and R. Reynolds, *Structural and Affective Aspects of Music From Statistical Audio Signal Analysis*, Journal of the American Society for Information Science and Technology, 57(11): 1526-1536, 2006

[4] K. Potter, G.A.Wiggins and M.T.Pearce, *Towards greater objectivity in music theory: Information-dynamic analysis of minimalist music*, In Musicae Scientiae 11(2), 295-322, 2007

[5] S. A. Abdallah and M. D. Plumbley, *Information dynamics: patterns of expectation and surprise in the perception of music*, Connection Science, Vol. 21, No. 2. (2009), pp. 89- 117.

[6] S. Dubnov, G. Assayag and A. Cont, *Audio Oracle: A New Algorithm for Fast Learning of Audio Structures*, In Proceedings of International Computer Music Conference (ICMC), September 2007.

[7] C. Allauzen, M. Crochemore, and M. Raffinot, *Factor oracle: A new structure for pattern matching*,In Conference on Current Trends in Theory and Practice of Informatics, pages 295310, 1999.

[8] G. Assayag, G.Bloch, A.Cont and S.Dubnov, *Interaction with Style*, In The Structure of Style: Algorithmic Approaches to Understanding Manner and Meaning, S. Argamon, K. Burns, S. Dubnov, (Eds.), Springer, 2010

[9] G. Strobl, G. Eckel and D. Rocchesso, *Sound texture modeling: A survey*, In Sound and Music Computing, 2006

[10] A.Lazier and P.Cook, *MOSIEVIUS: Feature driven interactive audio mosaicing*, In Proc. of Digital Audio Effects (DAFx) pp. 312317, London, September 2003.

[11] A. Lefebvre and T. Lecroq, *Compror: Compression with a Factor Oracle*, In Proceedings of the Data Compression Conference, 2001,

[12] J. Foote and M. Cooper, *Visualizing musical structure and rhythm via selfsimilarity*, In Proceedings of the International Computer Music Conference, pp. 419-422, 2001.
pp. 395-416

[13] Y. Weiss, *Segmentation using eigenvectors: a unifying view*, In Proceedings IEEE International Conference on Computer Vision p. 975-982 (1999)

[14] S. Dubnov, *Unified view of prediction and repetition structure in audio signals with application to interest point detection*, IEEE Transactions on Audio, Speech and Language Processing, 16(2), pp. 327-337, 2008.

[15] G.D. Birkhoff, *Aesthetic Measure*, Harvard Univ. Press, 1933.

[16] M. Bense, *Introduction to the Information-theoretical Aesthetics*, In Foundation and Application to the Text Theory, Rowohlt Taschenbuch Verlag, 1969 (In German).

[17] D.L.Baggi, G.Haus, *IEEE 1599: Music Encoding and Interaction*, IEEE Computer Vol. 42, No.3, pp. 84-87, 2009