Computational Measures of Musical Stimuli: The MUST toolbox

Ana Clemente, Manel Vila-Vidal, Marcus T. Pearce,
Germán Aguiló, Guido C. Corradi & Marcos Nadal

First, the package MIDI toolbox (https://github.com/MIDItoolbox/1.1) was used to import the MIDI files into MATLAB R2018a (The MathWorks, Inc., Natick, Massachusetts, USA) and to extract the onset (in beats of 0.5 s, from the beginning), the duration (also in beats of 0.5 s), and the MIDI pitch of each note event (Eerola & Toiviainen, 2004). Given a MIDI file of length T (4 s in our case) containing a total number of N notes, each note is denoted by its position in the sequence, i, that ranges from 1 to N. Then, for a given note i, we designated its pitch, onset time, and duration, as f_i, o_i, and d_i, respectively. Thereupon, we quantified the perceptual properties of the motifs as conceived in their design. High values mean high unbalance, jaggedness, asymmetry, and complexity.

The names of the functions of the MUST toolbox for MATLAB appear next to the measure's heading. Intermediate functions are also included in the toolbox, but not directly cited in the text. Although discarded for the empirical and theoretical reasons explained below, the functions of other measures developed in the investigation are also formulated in the MUST toolbox and cited in the text.

Balance

Bisect unbalance [`biUnbalance`]

Bisect unbalance quantifies how unevenly distributed the note events are between the two halves of the stimulus. For this purpose, we take into consideration onset times and disregard durations. We first find the distance between the first and the last onset times, $\tau = o_N - o_1$. We then find the proportion of note events with onset times either before or after $\tau/2$. Let them be called F_1 and F_2, respectively. Then, *bisect unbalance* is defined as:

$$b_1 = 1 - 4F_1F_2.$$

Note that this ensures that the measure assumes values in the range $[0, 1]$, with the maximum unbalance value (1) being attained either when $F_1 = 0$ or

$F_2 = 0$, and the minimum value (0) when there is a total balance between the two halves of the stimulus ($F_1 = F_2 = 0.5$).

Center of mass offset [comOffset]

The *center of mass offset* quantifies the distance between the temporal center of the stimulus ($\tau/2$, found as in *bisect unbalance*) and the center of mass of the distribution of note onsets across the stimulus. First, we take the average of all onset times to obtain what we call the center of mass (COM) of the stimulus. We then substract $\tau/2$ from the resulting value to find the distance between the two centers and finally divide by the total duration (τ) to obtain a normalized value. As we take the absolute value, the *center of mass offset* ranges between 0 and 1, with higher values reflecting a COM shifted further right or left from the temporal center of the stimulus. A value of exactly 0 reflects the totally balanced case in which the COM is found at the precise temporal center of the stimulus:

$$b_2 = \left| \frac{1}{\tau} \left(\frac{1}{N} \sum_{i=1}^{N} o_i - \frac{\tau}{2} \right) \right| = \left| \frac{1}{N} \sum_{i=1}^{N} \frac{o_i}{\tau} - \frac{1}{2} \right|.$$

Event heterogeneity [eventHeterogeneity]

This measure assesses how uniformly distributed the note events are, regardless of where the irregularities occur. First, we need to compute the local density curve. To do so, we consider sliding time windows of length $w = \alpha \frac{\tau}{N-1}$ with window step $\frac{w}{2}$ (i.e., with an overlap of 50%). Note that $\frac{\tau}{N-1}$ is the inverse of the global note onset density and, therefore, equal to the average inter-onset interval. Thus, the parameter α can be thought of as the expected number of notes per window (in our case, we used $\alpha = 2$). Then, we count the number of notes contained in each window j (with $j = 1, \cdots, J$; where J is the total number of sliding windows) and divide by the window length:

$$\delta_j = \frac{(N-1)n_j}{\alpha\tau},$$

with $n_j = \#\{\text{note events with onset time contained in window } j\}$. The local unbalance is defined as the ratio between the local density and the global density. As such, at time window j the local unbalance is simply $\frac{n_j}{\alpha}$. Segments with an accumulation (resp. depletion) of note events will have unbalance values larger (resp. smaller) than 1, which represents the case in which the local density coincides with the overall density. We finally take the standard deviation of this measure giving greater penalization to unbalanced events far from the center as compared to those close to it. The rationale behind this approach is that the accumulation of note events around the center may lead to a further release of tension that is not perceived as a lack of balance. *Event heterogeneity* is thus defined as:

$$b_3 = \frac{\sum_{j=1}^{J} \left(\frac{n_j}{\alpha} - 1\right)^2 w_j}{\sum_{j=1}^{J} w_j},$$

where the weight w_j ranges in the interval $[0, 1]$ and is the absolute time difference between the center of window j and the center of the stimulus $(\frac{\tau}{2})$ divided by $\frac{\tau}{2}$. This measure has no upper theoretical bound but, in practice, the largest value attained in the Balance subset is 1.411.

Contour

Average absolute interval [`avAbsInterval`]

This measure reflects the average absolute pitch interval on a logarithmic scale. It thus focuses on the magnitude of pitch changes between notes, one of the most prominent characteristics of a melodic profile:

$$c_1 = \frac{1}{N-1} \sum_{i=1}^{N-1} \log(|f_{i+1} - f_i| + 1).$$

The last term is incorporated to avoid values smaller than 1, for which the logarithm returns negative values. As a consequence, the minimum interval (the unison) is mapped to 0 and the output of the function monotonically increases with larger intervals. In the Contour subset, the largest value for this measure is 2.519.

Melodic abruptness [`melAbruptness`]

Melodic abruptness is a measure of intervallic sharpness, quantified as the accumulated intervallic size of changes of direction per time unit (as log-scaled intervals). To compute *melodic abruptness*, we first identify the notes where there is a change of direction (preceded by an ascending interval and followed by a descending interval, or vice versa) and define their sharpness as the average of the two surrounding intervals (log-scaled):

$$s_i = \log\left(\frac{|f_{i+1} - f_i| + |f_i - f_{i-1}|}{2} + 1\right).$$

Notes for which there is no change of direction as well as the first and the last notes of the stimulus are assigned zero sharpness. The *melodic abruptness* is then found by averaging the sharpness per time unit:

$$c_2 = \frac{1}{T} \sum_{i}^{N} s_i.$$

The largest value attained in the Contour subset is 11.387. Taking a visual metaphor, a change of melodic direction would mark the vertex, and the magnitude of the change per time unit would give the angle: the larger the intervals around a change of direction, the sharper the angle; and the greater the number of sharp changes per time unit, the more jagged the music.

Durational abruptness [`durAbruptness`]

A greater number of changes of melodic direction will create a more jagged contour (assuming the number and magnitude of intervals remain constant). *Durational abruptness* quantifies this effect as the proportion of the overall duration of the stimulus that is taken up with changes of melodic direction.

We first adjust the notes' durations so as to mimic perceptual features such as the smallest discriminable duration and the saturation of echoic memory after a certain threshold. To this end, we use Parncutt's phenomenological model, according to which the perceived "durational accent increases with IOI [inter-onset-interval] for small values of IOI and saturates as IOI approaches and exceeds the duration of the echoic store (auditory sensory memory)" (Parncutt, 1994: 427). In Parncutt's model, the durational accent is governed by the following equation:

$$\delta = \left[1 - \exp\left(\frac{d}{\theta}\right)\right]^i,$$

where d is the duration of the note (in seconds), θ accounts for the saturation duration (the larger θ, the sooner saturation is reached), and i flattens the curve close to $d = 0$ so as to account for minimum discriminable durations. As proposed by Parncutt, we set $\theta = 0.5$ and $i = 2$ in our implementation of the model. In what follows, we will refer to the sequence of durational accents adjusted by this model as δ_i.

Then, the *durational abruptness* of a stimulus is simply defined as:

$$c_3 = \frac{\sum_{\text{for i with change of direction}} \delta_i}{\sum_{i=1}^{N} \delta_i}.$$

The measure is bounded in the range [0,1].

Rhythmic abruptness [`rhythmAbruptness`]

This measure quantifies the average ratio between consecutive rhythmic figures, expressed as a quotient of durations. It thus focuses on rhythmic intervals, measuring how suddenly note durations change. Durations are first adjusted according to Parncutt's saturation model. Then, we compute the duration ratio for each pair of consecutive notes with the largest duration always in the numerator so as to ensure ratios are always larger than 1: $r_i = \delta_{i+1}/\delta_i$ if $\delta_{i+1} > \delta_i$, and $r_i = \delta_i/\delta_{i+1}$, otherwise. We finally take the average of all ratios across the whole stimulus:

$$c_4 = \sum_{i=1}^{N-1} r_i.$$

The largest value of *rhythmic abruptness* in the Contour subset is 8.615.

Symmetry

Total asymmetry [asymTotal]

Total asymmetry measures the accumulated pitch difference between original and reversed versions of the stimulus, quantified as follows. Let T be the total duration of the stimulus, and $f(t)$, the pitch at time t for all t between 0 and T. We define the instant asymmetry as:

$$a(t) = \left| f(t) - f(T-t) \right|.$$

Finally, the *total asymmetry* is defined as the average of this function across the whole motif:

$$s_1 = \frac{1}{T} \int_0^T a(t) dt = \frac{1}{T} \int_0^T \left| f(t) - f(T-t) \right| dt.$$

The *total asymmetry* has a lower bound (0), but it can grow arbitrarily large. However, in the Symmetry subset, the largest value is 6.645.

Asymmetry index [asymIndex]

Asymmetry index is also defined as the average of an instant asymmetry function across the whole motif but, in this case, the instant asymmetry is simply an index function that assesses whether there is a pitch difference between the motif played forwards and backwards, thus disregarding the specific pith distance. In particular, $a(t) = 1$ if $|f(t) - f(T-t)| > 0$, and 0 otherwise. Note that the asymmetry index will always be between 0 and 1. Its magnitude can be interpreted as the proportion of the overall duration of the stimulus for which it is asymmetric.

Complexity

Event density [eventDensity]

The basic principle of quantity of elements can be directly assessed as the total number of note events, only when comparing monophonic motifs with the same duration, like those of the set. In order to make the measure generalizable to other musical stimuli, total length must be controlled. In consequence, the density of events emerges as a better measure than quantity. *Event density* is then defined as the number of note events in the stimulus per time unit:

$$k_1 = \frac{N}{T}.$$

In the Complexity subset, the number of note events ranges from 3 to 31, and $T = 4$ s. Thus, the values for this measure in this subset range from 0.750 to 7.750.

Average local pitch entropy [avLocalp1entropy]

The pitch distribution might exhibit a certain degree of disorder at the whole stimulus level, while still preserving a certain local structure. To study this local effect, one can take small sliding windows (window length = 1 s, window step = 0.25 s in our case) and locally compute the pitch entropy in each window (see the next section for a detailed description of the *pitch entropy*). The average of this measure across the whole motif quantifies the local structure of the stimulus. In the Complexity subset, the maximum attained values is 1.526.

Pitch entropy [p1entropy]

This measure quantifies the entropy of the stimulus' pitch distribution. It disregards relations of pitches and durations, and hence any rhythmic or melodic structure. To compute the *pitch entropy*, we first need to characterize the pitch distribution. In order to do so, we count the number of times each pitch f appears in the stimulus:

$$P(f) = \frac{\#\{f_i = f \text{ for } i = 1, \ldots, N\}}{N}.$$

Note that here we are considering absolute pitch rather than pitch classes, and therefore C4 and C5, for example, will be considered as different elements. The *pitch entropy* is then defined as:

$$k_3 = -\sum_{\text{over all pitches } f \text{ appearing in the stimulus}} P(f) \log P(f). \tag{1}$$

This measure has a lower bound of 0, which corresponds to the case where only one pitch appears in the stimulus, regardless of how many times the note is repeated. As there is no theoretical limit to the number of different pitches than can appear in the stimulus, *pitch entropy* has no upper bound. In the Complexity subset, the maximum attained value is 2.316.

2-tuple pitch entropy [p2entropy]

2-tuple pitch entropy quantifies the entropy of the distribution of pairs of consecutive note pitches (i.e., a zeroth-order distribution of pairs of consecutive pitches). As with the *pitch entropy*, we first need to characterize the distribution of 2-note sequences (2-tuple pitch distribution henceforth). The probability of the pitch sequence (f, f') that appears at least once in the stimulus is defined as follows:

$$P((f,f')) = \frac{\#\{(f_i, f_{i+1}) = (f, f') \text{ for } i = 1, \ldots, N-1\}}{N-1},$$

where in the numerator we scan the whole stimulus for the tuple (f, f'), and every time there is a match we sum one unit for that sequence. Finally, the *2-tuple pitch entropy* is defined as the entropy of the 2-tuple pitch distribution. The maximum *2-tuple pitch entropy* attained in the Complexity subset is 3.063.

3-tuple pitch entropy [p3entropy]

This measure refers to the entropy of the distribution of sequences of 3 consecutive notes (i.e., pitches) or 3-tuple (f, f', f''). Similarly as before, we first define the 3-tuple pitch distribution as follows:

$$P((f,f',f'')) = \frac{\#\{(f_i, f_{i+1}, f_{i+2}) = (f, f', f'') \text{ for } i = 1, \ldots, N-2\}}{N-2},$$

and compute the entropy of this distribution. The maximum 3-tuple pitch entropy attained in the Complexity subset is 3.193.

2-tuple interval entropy [i2entropy]

The previous entropy measures reflect the degree of disorder of individual notes or groups of notes. However, research has shown that listeners represent melodies in memory in terms of more abstract representations of relative pitch structure such as melodic pitch interval (Dowling & Bartlett, 1981; Trehub, 1985). Therefore, it appears logic to transcend the pitch level and look at intervals. Indeed, we seem to perceive music in the most structured way possible (Snyder, 2009), attending to grouping principles (Bregman, 1990; Deutsch, 2013). We thus started by computing *interval entropy* [i1entropy], a more structured way to examine the entropy of 2-tuple distributions than *2-tuple pitch entropy*. However, the Pearson correlation with the behavioral assessment of musical complexity was markedly weaker ($r_p = .65$, $p < .05$) than that of 2-tuple pitch entropy ($r_p = .82$, $p < .001$). We therefore chose *2-tuple pitch entropy* for pitch distributions of 2 consecutive pitches over *interval entropy*.

Because melodic cells often contain more than 2 pitches, we addressed the entropy of sequences of 3 consecutive pitches. Indeed, *3-tuple pitch entropy* already comprises *pitch entropy*, *2-tuple pitch entropy*, and *interval entropy*. And *2-tuple interval entropy* is a more structured way of considering the entropy of tuples of 3 consecutive pitches. It measures the entropy of the distribution of sequences of 2 consecutive intervals. It is thus similar to *3-tuple pitch entropy* but differs in that sequences of 3 notes obtained from the same intervallic sequences are grouped together. Therefore, if the sequences C-D-E and G-A-B are considered two different elements in the 3-tuple pitch distribution, they are seen as the concatenation of two ascending major seconds in the 2-tuple interval distribution. To compute this, we first find the 3-tuple pitch distribution and collapse pitch tuplets that are obtained from the same sequence of intervals:

$$P(\bar{I}, \bar{I}') = \sum_{(f,f',f'') \text{ if } f+I=f' \text{ and } f'+I'=f''} P((f, f', f'')).$$

Then, we compute the entropy of this distribution. The maximum *2-tuple interval entropy* attained in the Complexity subset is 3.075. In motifs of duration, musical idiom, and methodological constraints like ours, sequences of more than 3 elements would be rare or not applicable to an important number of stimuli in the Complexity subset.

3-tuple duration entropy [d3entropy]

Regarding time structure, we followed the rationale applied to pitch distributions: We computed the entropy of single duration, of 2-tuple duration, and of 3-tuple duration distributions. As expected, the entropy of single durations [d1entropy] did not significantly correlate with perceived complexity ($p < .05$). This obeys to the restricted number of different durations (much more limited than that of pitches, especially in motifs like ours), and to the nature of the musical idiom and music perception (Pressing, 1999; Trehub, 1985). We therefore aimed to explore the existence of recursive rhythmical patterns across the stimulus. In musical motifs of short duration like these, the possibilities of imitative resources such as augmentation are limited. On the contrary, literal repetitions of rhythmic patterns are easier to implement and recognize, and help to apprehend the meter. Such patterns are typically formed by 2 or 3 rhythmic figures (e.g., medieval rhythmic modes). Therefore, we started by computing the entropy of 2-tuple duration distribution [d2entropy]. The Pearson correlation coefficients with the behavioral assessment resulted non-significant ($p > .05$), indicating that rhythmic sequences of two consecutive durations did not impact the perception of musical complexity in the Complexity subset, and we thus discarded this measure. The other possibility was to assess the entropy of sequences of 3 consecutive durations [d3entropy]. Now, the Pearson's r was significant ($p < .001$) and the entropy of 3-tuple duration distribution was included as a computational measure of perceived musical complexity.

As with all previous entropy-based measures, we first find the 3-tuple duration distribution by identifying all 3-note durational sequences appearing in the stimulus. Once the distribution is defined, computing the entropy is a straightforward operation (see equation 1). This measure has no theoretical upper bound but, in the Complexity subset, the maximum attained value is 2.185.

Weighted permutation entropy [wpEntropy]

Another way to assess the pitch complexity of the stimulus is to compute its permutation entropy, a measure that was initially designed to quantify the tendency to repeat ascending or descending n-element patterns (typically, $n = 3$)

within arbitrary time series (Bandt & Pompe, 2002). The permutation entropy is computed as the Shannon entropy of the distribution of order patterns of a time series. It thus takes into account only the order of the elements in the time series (whether there are ascents or descents) regardless of their absolute magnitude.

When sequences of $n = 3$ different elements are considered, six possible permutation patterns emerge: 123, 132, 213, 231, 312, and 321. However, when working with melodies, we should also account for pitch repetitions. This gives a total of 13 order signatures: the six permutations of three different elements (123, 132, 213, 231, 312, 321), the six possibilities with only two different pitches (122, 211, 112, 221, 121, 212) and one case in which all three elements have the same pitch (111).

Each order signature can represent different 3-element sequences, all of which, however, share a common ascending and descending pattern. Some examples should clarify how this association is performed: C4-D4-E4, C4-E4-F4, E4-G4-C5, and any sequence of strictly ascending pitches would correspond to the first pattern (123). The second pattern (132), on the other hand, represents any sequence in which the second pitch is higher than the first one, while the third one lies in between the first and the second one (e.g., C4-E4-D4, C4-B4-F4, G4-G5-C5). For the case of only two different pitches, the pattern 122, for example, represents any ascending interval followed by the repetition of the second pitch (e.g., C4-E4-E4, G5-C6-C6), while 212 represents any descending interval followed by the initial pitch (e.g., C4-B3-C4, G5-C5-G5). The last case (111) represents any sequence of three repeated pitches (e.g., C4-C4-C4, E5-E5-E5). As such, any 3-note sequence can be associated with a pitch order signature.

Then, the probability of a particular pitch order signature is computed as the number of times the order signature appears in the stimulus divided by the total number of 3-note groups in the stimulus ($N - 2$). For an order signature σ:

$$P(\sigma) = \frac{\#\{\text{pitch sequences } (f_i, f_{i+1}, f_{i+2}) \text{ with order signature } \sigma\}}{N-2}.$$

Once the probability distribution is found, the computation of the entropy is done as in equation 1 above. However, when computing the permutation entropy of a stimulus, two sequences of notes with equal permutation signature but different intervallic magnitude (C4-D4-E4 vs. C4-F4-B4) will have an equal impact on the computation of the permutation probabilities. A way to correct for this is to weight the effect of each 3-note sequence using the standard deviation of their pitch distribution (Fadlallah, Chen, Keil, & Príncipe, 2013; Xia, Shang, Wang, & Shi, 2016). In the case of the *weighted permutation entropy*, the following change is introduced: Each time the permutation signature is matched, we add the standard deviation of the pitch of the 3-note group (instead of 1), thus giving more weight to those groups where the intervals are larger:

$$P(\sigma) = \frac{\sum_{\text{for } (f_i,, f_{i+1}, f_{i+2}) \text{ with order signature } \sigma} SD(f_i, f_{i+1}, f_{i+2})}{N-2},$$

where SD stands for standard deviation.

The difference between the normal and the weighted version of the permutation entropy does not rely on the computation of the entropy from the probability distribution, but rather on the way the probability distribution is computed from the stimulus. As an example, imagine the sequence of notes C4-E4-G4-F4 (MIDI number: 60-64-67-65). When we use the unweighted version, we would say that the first three notes (C4-E4-G4) have an order signature of 123, while the second group (E4-G4-F4) has a permutation signature of 132. We have two of the 13 possible pitch order patterns, and each of them appears once, so they both have a probability of 0.5. However, in this particular case, the appearance of 123 is perceptually more salient with respect to 132, because C4-E4-G4 is perceived stronger or clearer than E4-F4-G4 due to their different intervallic magnitudes. The weighting accounts for this effect, as probabilities of appearance are corrected with the standard deviation of the pitch magnitudes. In the example, probabilities of encountering 132 or 123 are weighted as follows: The first three notes have permutation signature of 123 with a standard deviation of their pitches of 3.5, while the second group has a permutation signature of 132 with a standard deviation of their pitches of 1.5. Thus, we will assign probabilities of 0.7 and 0.3 to 123 and 132, respectively. The largest *weighted permutation entropy* attained in the Complexity subset is 2.178.

References

[1] Bandt, C., Pompe, B. (2002). Permutation entropy: a natural complexity measure for time series. *Physical review letters, 88*(17), 174102.

[2] Bregman A (1990). *Auditory scene analysis.* The MIT Press, Cambridge, MA.

[3] Deutsch, D. (2013). Grouping mechanisms in music. In *The Psychology of Music* (Third Edition) (pp. 183-248).

[4] Dowling, W. J., Bartlett, J. C. (1981). The importance of interval information in long-term memory for melodies. *Psychomusicology: A Journal of Research in Music Cognition, 1*(1), 30.

[5] Eerola, T., Toiviainen, P. (2004). *MIDI toolbox: MATLAB tools for music research.*

[6] Fadlallah, B., Chen, B., Keil, A., Príncipe, J. (2013). Weighted-permutation entropy: A complexity measure for time series incorporating amplitude information. *Physical Review E, 87*(2), 022911.

[7] Parncutt, R. (1994). A perceptual model of pulse salience and metrical accent in musical rhythms. *Music Perception: An Interdisciplinary Journal, 11*(4), 409-464.

[8] Pressing, J. (1999). Cognitive complexity and the structure of musical patterns. In *Proceedings of the 4th Conference of the Australasian Cognitive Science Society.*

[9] Snyder, B. (2009). Memory for music. *The Oxford handbook of music psychology*, 107-117.

[10] Trehub, S. E. (1985). Auditory pattern perception in infancy. In *Auditory development in infancy*(pp. 183-195). Springer, Boston, MA.

[11] Xia, J., Shang, P., Wang, J., Shi, W. (2016). Permutation and weighted-permutation entropy analysis for the complexity of nonlinear time series. *Communications in Nonlinear Science and Numerical Simulation, 31*(1-3), 60-68.