Mathematical Formulation of the `composer_algorithm` Music Generation Procedure

Overview

This document provides a compact mathematical description of the automatic music generation algorithm implemented in the MATLAB function `composer_algorithm`. The algorithm maps a time series of affective states in Russell's valence–arousal space to musical parameters (mode, rhythmic density, tempo, voicing, loudness), then generates MIDI note events by sampling from simple distributions.

Notation. MIDI pitches are integers (semitones) in $\mathbb{Z}$. The algorithm operates on discrete time indices $t \in \{1, \ldots, T\}$ where each t corresponds to a chord update (the MATLAB variable `idx`).

1 Inputs

Let the affective trajectory be

$$v[t] \in [0,1] \quad \text{(valence)}, \qquad a[t] \in [0,1] \quad \text{(arousal)}.$$

At each time step t, the algorithm reads $(v[t], a[t])$ and generates music for the current chord position in a 4-chord progression.

2 Harmonic material

2.1 Chord list

Define the chord list matrix $C \in \mathbb{Z}^{7\times 4}$:

$$C = \begin{bmatrix} 60 & 64 & 55 & 59 \\ 62 & 65 & 57 & 60 \\ 64 & 55 & 59 & 62 \\ 60 & 65 & 57 & 64 \\ 55 & 59 & 62 & 65 \\ 57 & 60 & 64 & 55 \\ 59 & 62 & 65 & 57 \end{bmatrix}.$$

Each row provides four MIDI pitch values used as a chord voicing.

2.2 Modes as 4-chord progressions

Define a 3-tensor $M[c, \tau, m] \in \mathbb{Z}$ with indices

$$c \in \{1,2,3,4\} \text{ (chord in progression)}, \qquad \tau \in \{1,2,3,4\} \text{ (tone in chord)}, \qquad m \in \{1,\ldots,7\} \text{ (mode)}.$$

Each mode m is defined by selecting four chord rows from C:

$$\begin{aligned}
\text{Mode 1 (Lydian)} &: [4, 7, 1, 4],\\
\text{Mode 2 (Ionian)} &: [1, 4, 5, 1],\\
\text{Mode 3 (Mixolydian)} &: [5, 1, 2, 5],\\
\text{Mode 4 (Dorian)} &: [2, 5, 6, 2],\\
\text{Mode 5 (Aeolian)} &: [6, 2, 3, 6],\\
\text{Mode 6 (Phrygian)} &: [3, 6, 7, 3],\\
\text{Mode 7 (Locrian)} &: [7, 3, 4, 7].
\end{aligned}$$

Concretely, if mode m uses chord rows $r_1, r_2, r_3, r_4 \in \{1, \ldots, 7\}$, then

$$M[c, \tau, m] = C[r_c, \tau] \quad \text{for } c = 1..4,\ \tau = 1..4.$$

Finally, all pitches are transposed down by 3 semitones:

$$M[c, \tau, m] \leftarrow M[c, \tau, m] - 3.$$

3 Affective mapping to musical parameters

At each time step t:

3.1 Mode selection from valence

The harmonic mode index is discretized from valence:

$$m(t) = 7 - \text{round}(6\, v[t]) \in \{1, \ldots, 7\}.$$

Thus high valence selects low mode indices (Mode 1), and low valence selects high mode indices (Mode 7).

3.2 Rhythmic roughness and density

The algorithm defines a roughness parameter

$$r(t) = 1 - a[t].$$

This parameter is used as a threshold that produces event gates; due to the construction below, the *probability of an event* equals $a[t]$.

3.3 Tempo (inter-onset time)

Define a per-substep pause duration (seconds):

$$\Delta(t) = 0.3 - 0.15\, a[t].$$

Hence $a[t] = 0$ yields $\Delta = 0.3$ s and $a[t] = 1$ yields $\Delta = 0.15$ s.

3.4 Voicing and bass register

Let $\nu(t) = v[t]$ denote the voicing control. The bass octave is chosen as

$$b(t) = \begin{cases} -12, & v[t] > 0.5, \\ -24, & v[t] \le 0.5. \end{cases}$$

3.5 Loudness (MIDI velocity range)

The maximal note velocity is quantized from arousal:

$$L_{\max}(t) = 60 + 40 \cdot \frac{\mathrm{round}(10a[t])}{10} \in [60, 100].$$

A global minimal velocity is

$$L_{\min} = 50.$$

Each note velocity ℓ is sampled as an integer uniform random variable:

$$\ell \sim \mathrm{UnifInt}\{L_{\min}, \dots, L_{\max}(t)\}.$$

4 Stochastic rhythm generation

For each chord at time t, the algorithm samples two independent 8-step activation vectors:

$$\boldsymbol{g}^{(1)}(t) = (g_1^{(1)}, \dots, g_8^{(1)}), \qquad \boldsymbol{g}^{(2)}(t) = (g_1^{(2)}, \dots, g_8^{(2)}),$$

constructed by thresholding i.i.d. uniform samples:

$$U_k^{(j)} \sim \mathrm{Uniform}(0,1), \qquad g_k^{(j)}(t) = \mathbb{I}\left[U_k^{(j)} \geq r(t)\right], \quad j \in \{1,2\},\ k \in \{1, \dots, 8\}.$$

Since $r(t) = 1 - a[t]$,

$$\Pr\big(g_k^{(j)}(t) = 1\big) = \Pr\big(U \geq 1 - a[t]\big) = a[t],$$

so

$$g_k^{(j)}(t) \sim \mathrm{Bernoulli}\big(a[t]\big) \quad \text{i.i.d.}$$

The expected number of triggered micro-events per gate vector is

$$\mathbb{E}\left[\sum_{k=1}^{8} g_k^{(j)}(t)\right] = 8a[t].$$

5 Stochastic "brightness" (octave shifts) from valence

For each chord, the algorithm samples six octave-shift controls

$$B_i(t) \in \{-1, 0, 1\}, \quad i = 1, \dots, 6,$$

based on valence. Let $U_i \sim \mathrm{Uniform}(0,1)$ i.i.d.

Case 1: $v[t] < 0.5$ (downward bias)

$$B_i(t) = \begin{cases} -1, & U_i > 2v[t], \\ 0, & U_i \leq 2v[t]. \end{cases}$$

Hence

$$\Pr(B_i = -1) = 1 - 2v[t], \qquad \Pr(B_i = 0) = 2v[t], \qquad \Pr(B_i = +1) = 0.$$

Case 2: $v[t] \geq 0.5$ (upward bias)

$$B_i(t) = \begin{cases} +1, & U_i < 2(v[t] - 0.5) = 2v[t] - 1, \\ 0, & U_i \geq 2v[t] - 1. \end{cases}$$

Hence

$$\Pr(B_i = +1) = 2v[t] - 1, \qquad \Pr(B_i = 0) = 2 - 2v[t], \qquad \Pr(B_i = -1) = 0.$$

These values are applied as octave shifts in semitones via $12B_i$.

6 Note generation per chord

Let the current chord-in-progression index be $c \in \{1, 2, 3, 4\}$ for the progression of mode $m(t)$. Define the three principal chord tones:

$$p_1(t) = M[c, 1, m(t)], \quad p_2(t) = M[c, 2, m(t)], \quad p_3(t) = M[c, 3, m(t)].$$

6.1 Sustained chord tones

The algorithm triggers note-on events (duplicated across MIDI channels 1 and 2) for:

$$p_1(t) + 12B_1(t), \qquad p_2(t) + 12B_2(t), \qquad p_3(t) + 12B_3(t),$$

with independent velocities $\ell \sim \mathrm{UnifInt}\{L_{\min}, \ldots, L_{\max}(t)\}$ per event.

6.2 Bass note

On MIDI channel 3, a bass note is triggered:

$$p_{\mathrm{bass}}(t) = p_1(t) + b(t), \quad b(t) \in \{-12, -24\},$$

with velocity sampled from the same integer-uniform range.

6.3 Eight-step embellishment loop

For microsteps $k = 1, \ldots, 8$:

- If $g_k^{(1)}(t) = 1$, play an additional note based on the root:
$$p_1(t) + 12B_5(t).$$
- If $g_k^{(2)}(t) = 1$, choose a chord tone index $J_k \sim \mathrm{UnifInt}\{2, 3\}$ and play:
$$M[c, J_k, m(t)] + 12B_6(t).$$
- Wait $\Delta(t) = 0.3 - 0.15a[t]$ seconds before the next microstep.

All embellishment notes are emitted on MIDI channel 1 with velocities sampled i.i.d. as above.

7 Compact generative process

For each chord time step t with chord position c:

1. Read affective input $(v[t], a[t])$.
2. Compute controls:
$$m(t) = 7 - \mathrm{round}(6v[t]), \quad \Delta(t) = 0.3 - 0.15a[t], \quad L_{\max}(t) = 60 + 40\frac{\mathrm{round}(10a[t])}{10}, \quad b(t) \in \{-12, -24\}.$$
3. Sample gates:
$$g_k^{(1)}(t), g_k^{(2)}(t) \overset{\text{i.i.d.}}{\sim} \mathrm{Bernoulli}(a[t]), \quad k = 1, \ldots, 8.$$
4. Sample octave shifts $B_1(t), \ldots, B_6(t)$ from the piecewise valence-dependent distribution.
5. Emit sustained chord tones p_1, p_2, p_3 with octave shifts, plus bass p_{bass}.
6. For $k = 1..8$, emit embellishment notes conditioned on $g_k^{(1)}, g_k^{(2)}$, waiting $\Delta(t)$ between microsteps.

8 Immediate implications

- **Rhythmic density is linear in arousal.** For each 8-step gate vector,

$$\mathbb{E}\left[\#\text{events}\right] = 8a[t].$$

  Two independent vectors yield an expected $16a[t]$ triggered micro-events per chord (ignoring coincident events).

- **Valence controls harmony & register.** Valence discretizes the mode $m(t)$ and biases octave-shift random variables $B_i(t)$ (downward for $v < 0.5$, upward for $v > 0.5$), and also chooses bass register.

Reference for algorithm usage:

Ehrlich, S. K., Agres, K. R., Guan, C., & Cheng, G. (2019). A closed-loop, music-based brain-computer interface for emotion mediation. *PLOS ONE*, 14(3), e0213516. `https://doi.org/10.1371/journal.pone.0213516`