# On VAE Latent Space Vectors Distributed Evolution Driven Music Generation

Gleb Rogozinsky[1][0000−0001−5698−2347] and Aleksei Shchekochikhin[2][0000−0003−2603−9133]

[1] The Bonch-Bruevich Saint-Petersburg State University of Telecommunications
gleb.rogozinsky@gmail.com
[2] ashekochikhin@gmail.com

**Abstract.** Nowadays, the deep learning technics used in the algorithms of computer music generation are rapidly developing. One of the common problems of the deep learning-based methods is the successful reproduction of music score sequences within the short time intervals as well as on long ones. To solve it, one may use so-called hierarchical models, which typically lead to a computational load. Earlier systems based on evolutionary computation methods, especially genetic algorithms, had already proven their ability to model variations of musical sequences within the long time scales. The purpose of the presented study is to demonstrate how to incorporate deep learning models, evolutionary computations, and rule-based computer music generation technics to approach the most effective features of each one technic correspondingly. The paper presents the combined method of music sequence generation based on a distributed genetic algorithm with the Variational Autoencoder(VAE) genotype-to-phenotype mapping. The research includes experimental investigation of the latent space vectors evolutionary driven dynamics, to figure out the ways of how one can achieve the state of a controllable evolution process. The paper also presents the set of rule-based approaches to modify system-generated music sequences to meet predictable high perceptual differences at the low computational costs. The outcomes of the genetic algorithm runs at the various numbers of agents and different mutations are given, together with several examples of music scores, generated within the research.

**Keywords:** Algorithmic Music Generation · Computer Music Technics · Genetic algorithm · VAE



## 1 Introduction

Algorithmic music generation, also known as algorithmic music, has a long history of development, originated in the days of Mozart to undergo the rapid

progress due to co-evolution with the informational and communicational technologies. Comparing to the days of Illiac Suite by ILLIAC I (Illinois Automatic Computer) mainframe coded by Lejaren Hiller in 1957, the state-of-art software allows user to have his/her own pocket mobile 'Illiac' running on iOS/Android. Meanwhile, the computational advantage still not able to reach the singularity frontier for the computer music systems, mostly because of the simplicity-complexity dualism of human-generated music. Human music can be simple to its extreme, but at some particular moment, one little pause, or unexpected modulation can turn several measures into the masterpiece. The authors' present study continues their previous research in the field of distributed computer music generation systems, aimed to set of competing agents, for imitation of human-featured process of selection between several creative thoughts.

## 2 Music generation technics comparison

Computer music generation remains one of the problems of machine creativity. Since 1960s, applied technics has been evolved from simple rule models and Markov chains [7] to complex ones, based on evolutionary computations [2], [9], and deep learning models [18], [19], [20]. Still, there is no universal way to compare different computer-aided generative technics. One of the methods to value the results of generated music pieces is the Music Turing Test, one of its first usages was described in [6]. The authors conducted a research to compare the existing computer music generation technics, to figure out the most successful ones. Fifty respondents were asked to listen to 12 different short music pieces, 10 of which were generated by computer algorithms, and to guess which one was composed by human being. Table 1 gives a summary of the comparison of music composition technics. There are some other approaches to evaluate the quality of computer-generated music pieces. In [8] peers were asked to compare generated samples by raking them in a range from 1 to 10. 1 grade corresponded to "completely random noise" while a 10 grade corresponded to "composed by a novice composer". In [17] authors indicated the main drawback of such quality evaluation technics, which are high costs and low reproducibility, and shown how objective quantities, such as model entropy and mutual information as a function of the distance between two notes, may be used for computer music generation technic outcomes evaluation.

**Table 1.** Turing test for music composition techniques

| Music composition technique | Mean Turing test pass, % |
|---|---|
| Composed by human | 69 |
| Deep learning | 60 |
| Complex technics | 56 |
| Genetic algorithm | 52 |
| Markov chains | 46 |
| Cellular automata | 30 |

Nevertheless, while technics based on deep learning models appear to be most advanced, there some known limitations of the usage of them:

- deep learning methods do not actually compose a new piece of music, but rather produce a sequence of notes, which is statistically equal with the given dataset; thus the new genre could not appear in such generation model;
- complex hierarchical models are to be applied to achieve good results for generation pieces in long time ranges [12];
- the process of learning as well as generation still needs a lot of computations, but the effectiveness of recent models increases [4].

On the other hand, evolutionary computation-based methods are proven to provide a way to model music melody variations on long time scales [1].

## 3 Latent space vectors evolution generative model

In [13], we have proposed the method of music generation with a distributed genetic algorithm. One of the main difficulties in genetic algorithm music generation system design is to define genotype-to-phenotype mapping as well as define a fitting function. In [11], one may find a rich survey of different approaches for evolutionary computer music generation systems design.

It has been shown that VAEs and GANs can be used to perform the genotype-to-phenotype mapping, performing the interactive evolutionary computation for image generation[3] as well as the 2 bar music sequences[15].

We propose the following technique for infinity personalized music generation, based on the genetic evolution of latent vectors (see figure 1):

- the distributed genetic algorithm is used as an attractor for the music generation; each agent in the system is associated with the one user; all agents together form the population for the genetic algorithm;
- VAE is used for the genotype-to-phenotype mapping;
- each agent has a target state, the distance to that is used as the fitness function;
- the agent target states are set by users, mapped to the natural language descriptions;
- at each system run loop step, agents obtain the states of others to perform the next iteration of the genetic algorithm.

The current version of the system model is available online[3]. MusicVAE autoencoder[12] from Magenta was used to perform the genotype-to-phenotype mapping to decode genetic algorithm run results into note sequences. MusicVAE encoder is a two-layer bidirectional LSTM network. MusicVAE decoder is a hierarchical RNN. The key assumption of implementing genotype-to-phenotype mapping with the VAE model is to achieve perceptual continuity: we assume that small distances in latent space will correspond to perceptually-close decoded music sequences.

[3] https://github.com/ashekochikhin/aGASim

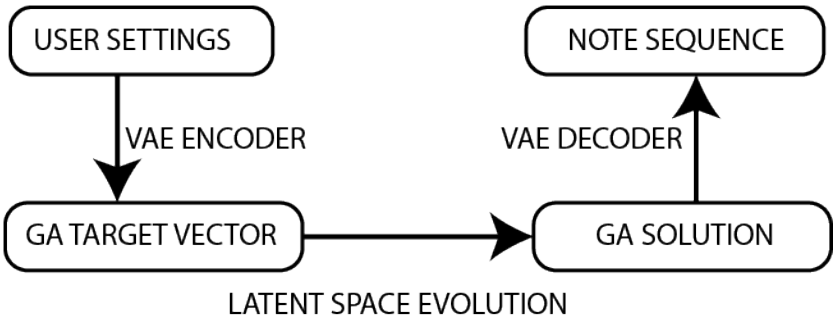


**Fig. 1.** Distributed Genetic Algorithm Music Generation with VAE Genotype-to-Phenotype Mapping

In the experiments, the following genetic algorithm realization was implemented:

- agent get current states of all other agents in the system;
- the Hamming distance closest to the current target state agent is determined;
- single point crossover runs at current and closest agents providing two new children candidates;
- system specified number of random mutations runs;
- closest to the target state candidate replaces current agents state.

The model system has the following variable parameters:

- $N$ - number of agents;
- $l_{ch}$ - chromosome length;
- $p_m$ - number of mutations after crossover;
- $l_\Sigma$ - gen's alphabet length.

Figure 2 shows the influence of mutations probability on the median population distance of current agents states to the target ones. We denote constant asymptote of the distance to the target state as stable distance. With the growth of the population size influence of random mutation decreases, see figure 3. For a given population size stable distance to target state $D_{st}$:

$$D_{st} \sim l_{ch}(1 - e^{-c_1(p_m/l_{ch}+c_2)}), \tag{1}$$

where $c_1$ and $c_2$ are some constants defined by system configutaion.

Figure 4 and 5 show the influences of mutation probability, and the number of agents on the median distance of current agents states to the target ones for different chromosome lengths. With the growth of chromosome length number of agents required to achieve relatively close stable distance to the target increases.

Figure 6 shows the decay of the distance to the target state with the agents' population size. For a given chromosome length stable distance to target state

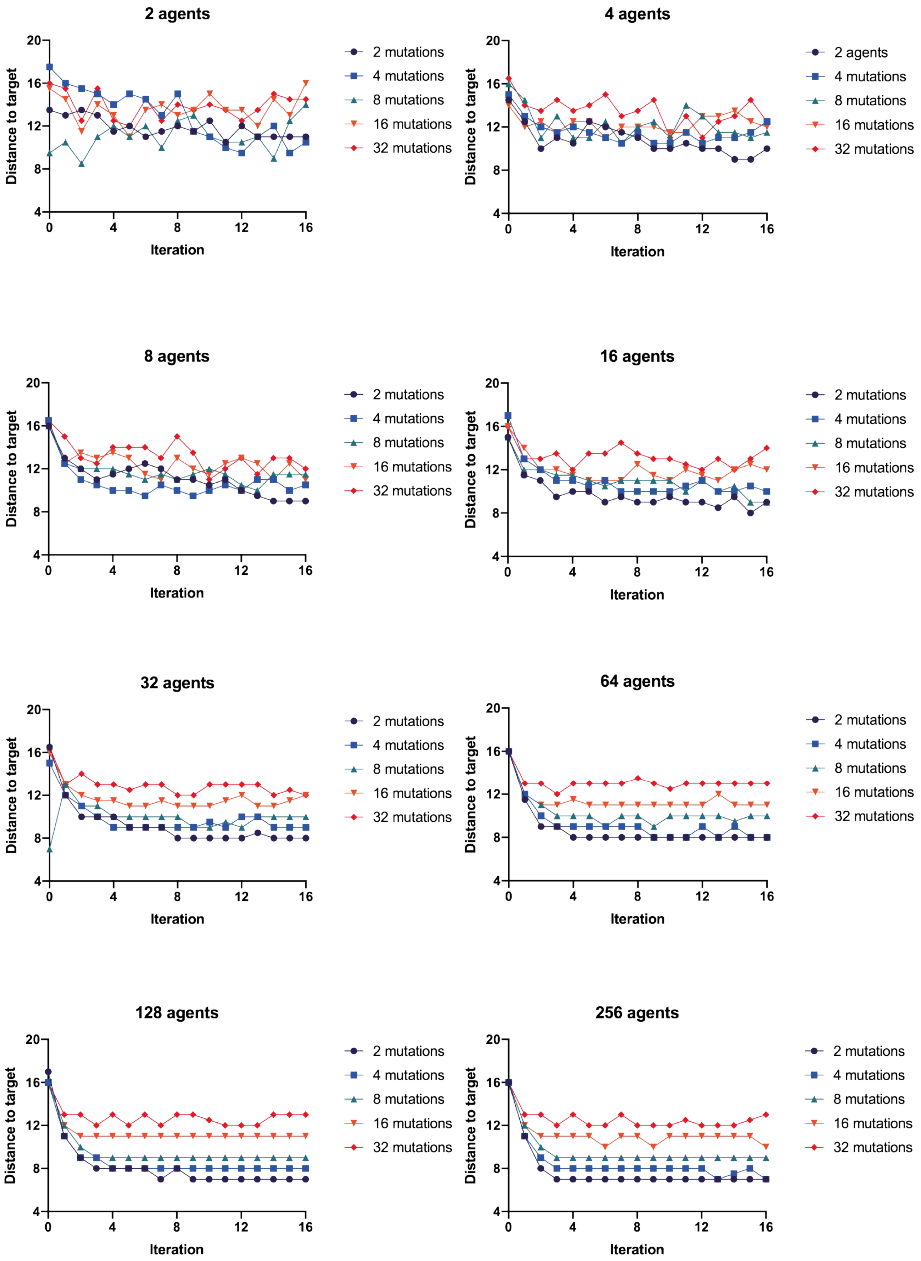


**Fig. 2.** Examples of modeling distributed GA run. Chromosome length is equal to 32. Alphabet length is equal to 2. With the growth of the population, more agents, current solutions become closer to the target vectors.

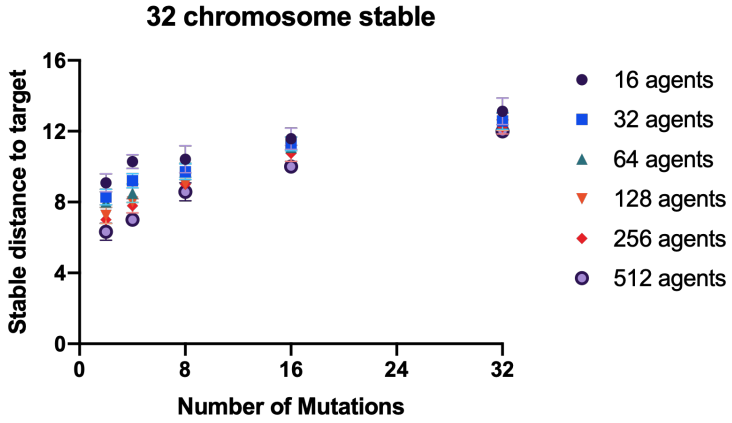


**Fig. 3.** The dependency of stable distance to the target on the number of mutations. Chromosome length is equal to 32. Alphabet length is equal to 2..

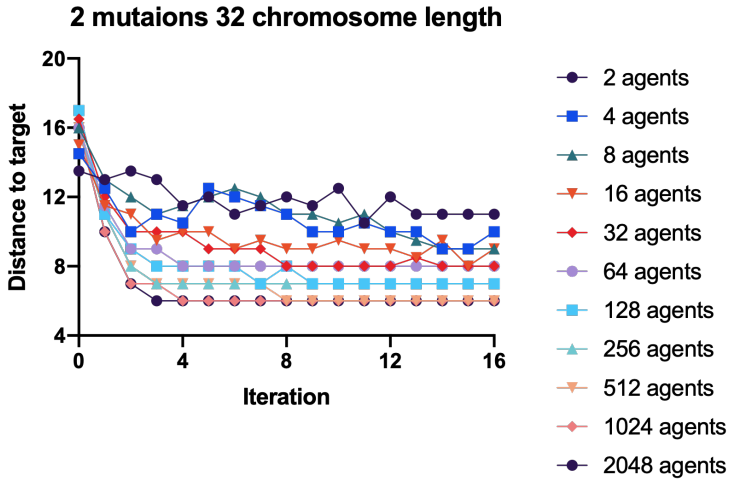


**Fig. 4.** Examples of modeling distributed GA run. Chromosome length is equal to 32. Alphabet length is equal to 2. Mutations influences higher on smaller populations.

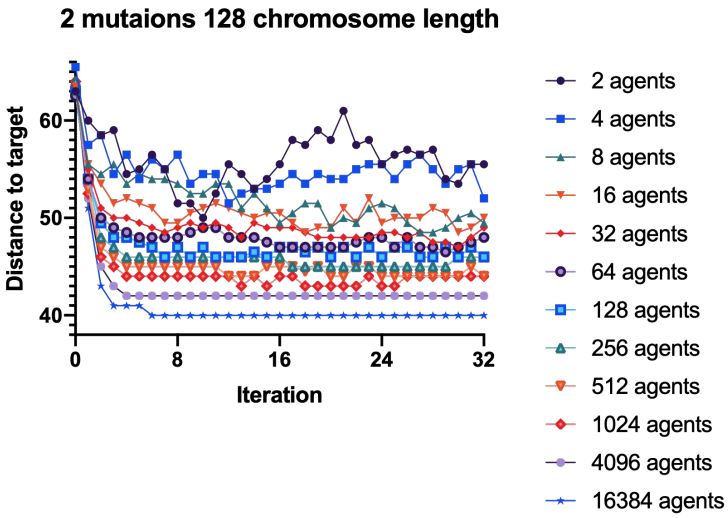


**Fig. 5.** Examples of modeling distributed GA run. Chromosome lengths is equal to 128. Alphabet length is equal to 2. Mutations influences higher on smaller populations.

may be approximated with an inhibitory dose-response curve, $D_{st}$:

$$D_{st} \sim \frac{l_{ch}}{1 + (\frac{N}{c_2})^{c_1}}, \tag{2}$$

where $c_1$ and $c_2$ are constants defined by system configutaion.

Figure 7 shows the growth of the stable distance to the target state with the alphabet length. For a given population size stable distance to target state $D_{st}$:

$$D_{st} \sim l_{ch}(1 - e^{-c_1 l_\Sigma}), \tag{3}$$

where $c_1$ is a constant defined by system configutaion.

Provided analysis of the distributed genetic algorithm behavior on system variable parameters is to be used to keep the correct run. As the genetic algorithm is used not for optimization but generation, it should always be close enough to the extremum but never stack on it. Random mutations may be used to prevent continuous repeats of every solution. If there are not enough agents to be close enough to the target system may generate virtual ones.

## 4 Discussion

Figure 8 gives an example of a generated sequence of the 16 bar length. Note that implemented VAE was trained to produce the 2 bar length sequences. One can notice that the presented sequence is in the key of B-flat minor. The melody generally remains within the selected tonality. We must admit that with the

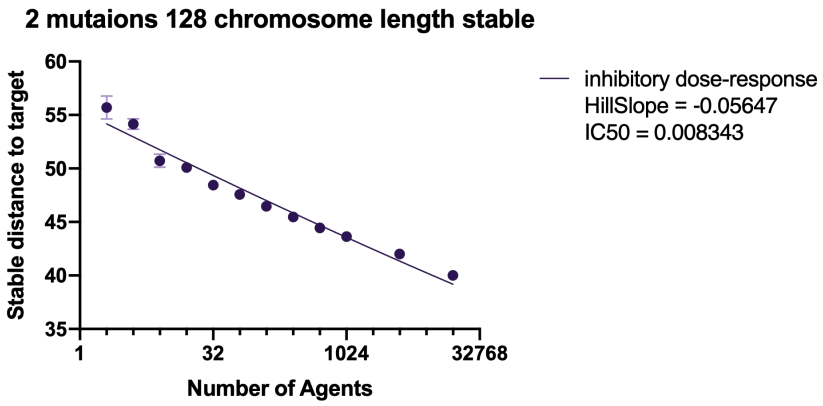


**Fig. 6.** Example of modeling distributed GA run. The chromosomes' lengths are equal to 128. The alphabet length is equal to 2. The dependency of stable distance to the target states on population size.

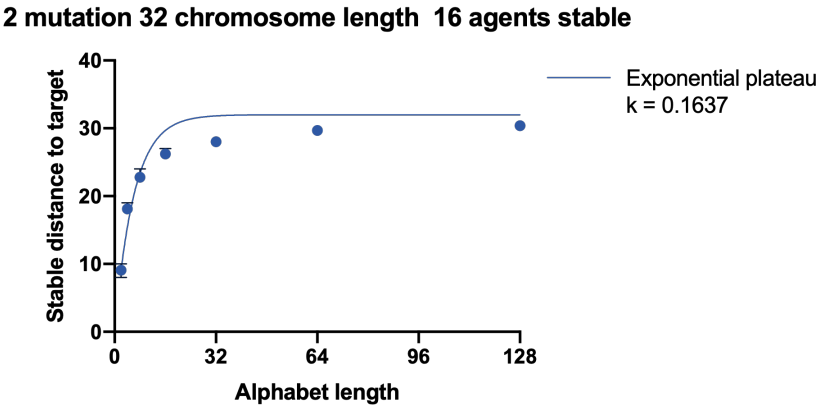


**Fig. 7.** Example of modeling distributed GA run. The chromosomes' lengths are equal to 32. The population size is equal to 16 agents. Dependence of stable distance to the target states on the gen alphabet length.

described approach on the long timescales (32 to 64 bars) system 'losses' memory of the initial sequence and its attributes.

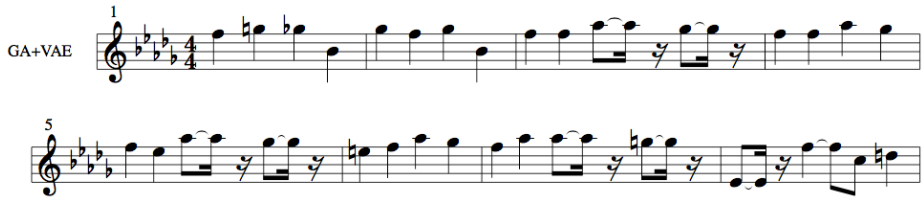


**Fig. 8.** Example of generating of the 16 bar note sequence

To get better scale stability, one may apply some algorithmic constraints to the generated sequences. In [5] was shown how the constraints in the latent space might be used to reach a better successful generation rate. We assume that restrictions in the phenotype space are also practical. For example, one may use major scale constraint to the mentioned sequence and get one presented in figure 9. To apply scale constraint, one should first determine the sequences key. We provide a simplified algorithm to identify the key of the sequence by giving a score to each note for each music key:

- +2, if a note from the sequence is I, IV, or V scale degree;
- +1, if a note from the sequence is any other scale degree;
- −1, if a note from the sequence is out of a scale;
- if the total scores of the sequence are equal for two different scales, the one with fewer symbols is to be chosen.

After the original scale is determined, the sequence may be quantized to the desirable one by replacing out of scale notes with the closest one from the scale. There are also more complex algorithmic technics to determine the original sequence's key [16], as well as deep learning-based approaches for key extraction from raw audio samples [10].

Another constraint application is a rhythmic modification. In spite of original VAE was trained to decode latent space vectors to a sequence in four-four time signature, the generated sequence can also be easily quantized to three-four time, see figure 10, or six-eight time, see figure 11.

Rhythmic transformations may also be performed according to statistically motivated rules. Depending on the music genre, different rhythmic sequences - rudiments - are more or less common [14]. Applying rudimental rhythmic filtering will result in not only statistically right results but in the generation of sequences, which may be comfortably performed by a human.

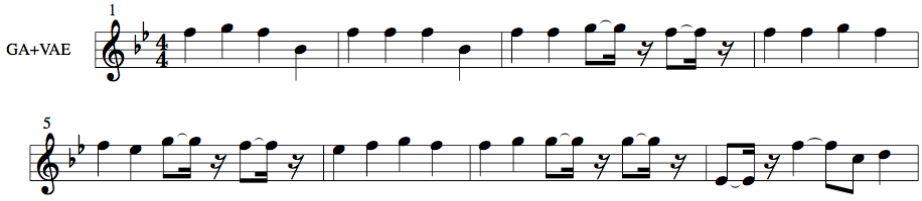


**Fig. 9.** The generated sequence with applied major scale constraint

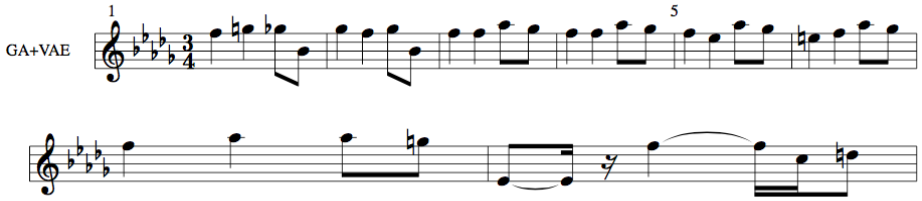


**Fig. 10.** The generated sequence in three-four time

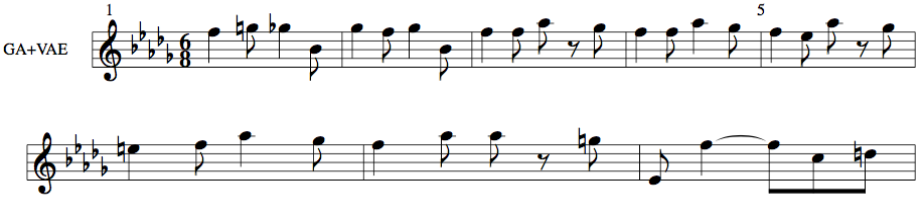


**Fig. 11.** The generated sequence in six-eight time

Therefore, one of the approaches to improve nowadays' most advanced deep learning-based music sequences generation techniques is to combine them with the well- known algorithmic solutions such as genetic algorithm and rule-based constraints. This approach leads to consistent results with a low computation load.

## 5 Conclusion

The approach that has been discussed in the paper leads us to further study of the proposed distributed genetic algorithm with VAE genotype-to-phenotype mapping music generation model. The combining technics of algorithmic music proved to play a key role in the effective composition. Different approaches possess their peculiar strong and weak sides, thus suitably combining them will produce a compositionally attractive outcome. Generally speaking, an agent-based approach with competing 'ideas' can be treated as a model of world music evolution, i.e. starting from the basic simple sequences, it evolved into very complex, polytonal, and even atonal structures. There are already some popular applications and online services existing. We can assume that their effectiveness both in the sense of speed and music 'ideas' is also due to a proper combined approach. The application area of such services allows us to estimate the future of such systems. Various recreational zones, both private and public, with ability to be adjusted, i.e. to the weather conditions, daytime, team progress, wearable device data etc. In addition, as a set of assistive technologies for the human composers, virtual reality randomization effects etc. One of the greatest features of such systems is their ability to be individualized according to the client data.

## References

1. Arutyunov, V., Averkin, A.: Genetic algorithms for music variation on genom platform. Procedia Computer Science **120**, 317 – 324 (2017). https://doi.org/https://doi.org/10.1016/j.procs.2017.11.245, http://www.sciencedirect.com/science/article/pii/S1877050917324572, 9th International Conference on Theory and Application of Soft Computing, Computing with Words and Perception, ICSCCW 2017, 22-23 August 2017, Budapest, Hungary
2. Biles, J.: Genjam: A genetic algorithm for generating jazz solos (07 1994)
3. Bontrager, P., Lin, W., Togelius, J., Risi, S.: Deep Interactive Evolution. arXiv e-prints arXiv:1801.08230 (Jan 2018)
4. Engel, J., Agrawal, K.K., Chen, S., Gulrajani, I., Donahue, C., Roberts, A.: GANSynth: Adversarial neural audio synthesis. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=H1xQVn09FX
5. Engel, J., Hoffman, M., Roberts, A.: Latent constraints: Learning to generate conditionally from unconditional generative models. In: International Conference on Learning Representations (ICLR) (2018), https://openreview.net/pdf?id=Sy8XvGb0-
6. Hadjeres, G., Pachet, F., Nielsen, F.: Deepbach: a steerable model for bach chorales generation. In: ICML (2016)

7. Hiller, L.A., Isaacson, L.M.: Experimental Music; Composition with an Electronic Computer. Greenwood Publishing Group Inc., Westport, CT, USA (1979)
8. Huang, A., Wu, R.: Deep learning for music. CoRR **abs/1606.04930** (2016), http://arxiv.org/abs/1606.04930
9. II, R.: Composing with Genetic Algorithms: GenDash, pp. 117–136 (01 2007)
10. Korzeniowski, F., Widmer, G.: End-to-end musical key estimation using a convolutional neural network (06 2017)
11. Miranda, E.R., Biles, J.A.: Evolutionary Computer Music. Springer-Verlag, Berlin, Heidelberg (2007)
12. Roberts, A., Engel, J., Raffel, C., Hawthorne, C., Eck, D.: A hierarchical latent vector model for learning long-term structure in music. In: International Conference on Machine Learning (ICML) (2018), http://proceedings.mlr.press/v80/roberts18a.html
13. Rogozinsky G., S.A.: The distributed system model for evolutionary generation of audio content. INFORMATION TECHNOLOGIES AND TELECOMMUNICATIONS **2**(2), 20–26 (2014)
14. Sethares, W.: The geometry of musical rhythm: what makes a "good" rhythm good? Journal of Mathematics and the Arts **8**, 135–137 (12 2014). https://doi.org/10.1080/17513472.2014.906116
15. Shafkat, I.: Music by means of human selection (Mar 2019), https://towardsdatascience.com/music-by-means-of-natural-selection-11934d7e89a3
16. Shmulevich, I., Yli-Harja, O.: Localized key-finding: Algorithms and applications. Music Perception: An Interdisciplinary Journal **17** (07 2000). https://doi.org/10.2307/40285832
17. Tikhonov, A., Yamshchikov, I.P.: Music generation with variational recurrent autoencoder supported by history. CoRR **abs/1705.05458** (2017), http://arxiv.org/abs/1705.05458
18. van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K.: WaveNet: A Generative Model for Raw Audio. arXiv e-prints arXiv:1609.03499 (Sep 2016)
19. Vasquez, S., Lewis, M.: MelNet: A Generative Model for Audio in the Frequency Domain. arXiv e-prints arXiv:1906.01083 (Jun 2019)
20. Yang, L.C., Chou, S.Y., Yang, Y.H.: MidiNet: A Convolutional Generative Adversarial Network for Symbolic-domain Music Generation. arXiv e-prints arXiv:1703.10847 (Mar 2017)