Automatic Music Synthesizer using AI

Harsh Bansal   
Department of Information Technology

SSN College of Engineering Chennai,India  
[harsh2110303@ssn.edu.in](mailto:harsh2110303@ssn.edu.in)

Dr.N Radha

Associate Professor,

Department of Information

Technology

Harsha Nandhini K M   
 Department of Information Technology

SSN College of Engineering Chennai,India  
[harshanandhini2110106@ssn.edu.in](mailto:harshanandhini2110106@ssn.edu.in)

Harshini D   
Department of Information Technology

SSN College of Engineering Chennai,India  
[harshini2110770@ssn.edu.in](mailto:harshini2110770@ssn.edu.in)

Dr.R.Swathika

Associate Professor ,

Department of Information Technology

*Abstract*-- **The "Automatic Music Synthesizer Using AI" introduces a novel approach to the automated generation of musical compositions through the application of genetic algorithms (GAs). The system employs GAs to evolve and refine musical melodies, incorporating objective criteria for beauty, harmonics, and rhythmic structure. The generated compositions exhibit intervals that are pleasing to the human ear, possess understandable rhythms, and can be subtly modified for pleasant variations. The study demonstrates the effective regulation of composition quality and shape, leveraging coding techniques to control tones, intervals, and rhythm. The algorithm's flexibility allows users to define specifications and reference persons, influencing the selection and creation of compositions This project represents a creative fusion of music theory and computational intelligence, offering a promising tool for musicians and composers seeking automated and inspiring music generation.**

Keywords—music composition, fitness, genome, genetic algorithm, MIDI

# Introduction

The "Automatic Music Synthesizer Using Genetic Algorithm" is a groundbreaking project at the intersection of music composition and artificial intelligence. Traditional music composition can be a time-consuming and intricate process, requiring a deep understanding of music theory and significant creative input. This project addresses the challenges faced by both novice and experienced composers by introducing an innovative approach that leverages genetic algorithms to automate parts of the music production process.

Traditional synthesis approach often demands laborious manual input and intricate parameter adjustments, making the composition process both technical and time-consuming. The proposed music synthesizer, leveraging a genetic algorithm, seeks to revolutionize this methodology by introducing a computational model inspired by the principles of natural evolution. Grounded in Charles Darwin's theory of natural selection, the genetic algorithm iteratively generates, evaluates, and refines musical compositions. This automated process aims to produce compositions that align with predefined criteria for beauty, harmonics, and rhythm.

The algorithm dynamically regulates various aspects of composition quality, offering a flexible and user-friendly platform for creating unique and aesthetically pleasing musical pieces. By integrating genetic algorithms into music synthesis, the proposed system aims to streamline the composition process and provide a more intuitive and efficient means for musicians to realize their creative vision.

The motivation behind the development of the "Automatic Music Synthesizer Using AI" stems from a desire to explore the intersection of music theory and cutting-edge computational intelligence. Traditional approaches to music composition often rely on the expertise and creativity of human composers. However, with the advancements in artificial intelligence, there is a unique opportunity to leverage algorithms to automate and enhance the creative process. This project is motivated by the belief that the marriage of music and AI can open new possibilities for artistic expression, offering musicians and composers a tool that not only accelerates the composition process but also introduces innovative and inspiring musical outcomes.

The motivation here lies in exploring whether an algorithmic approach can not only meet objective criteria such as beauty, harmonics, and rhythmic structure but also capture the subjective nuances that make music a deeply personal and emotional experience. This exploration reflects a curiosity to understand how computational methods can contribute to the richness and diversity of musical expression.

In addition to the core conceptualization and implementation, the paper will explore potential enhancements based on the initial success of the system. These include discussions on the development of a user-friendly graphical interface for better interaction, automated fitness evaluation methods, real-time playback features, advanced music theory integration, and parameter tuning options for users to customize the genetic algorithm.

# RELATED WORK

In paper [1] on Genetic Algorithms in Search, Optimization and Machine Learning states, Genetic algorithms are probabilistic search procedures designed to work on large spaces involving states that can be represented by strings. These methods are inherently parallel, using a distributed set of samples from the space (a population of strings) to generate a new set of samples. Although there are several different types of genetics-based machine learning systems, this article [1] concentrates on classifier systems and their derivatives.

In paper [2] “Music Composition with Interactive Evolutionary Computation,” in Proc. Generative Art2000, the 3rd International Conference on Generative Art [2] Describes a new approach to the music composition, more precisely the composition of rhythms, using IEC. The main feature of our method is to combine genetic algorithms (GA) and genetic programming (GP). In the [2] system, GA individuals represent short pieces of rhythmic patterns, while GP individuals express how these patterns are arranged in terms of their functions. Both populations evolved interactively through the user's evaluation.

In paper [3] on the GP- Music System: Interactive Genetic Programming for Music Composition Created automatic fitness raters based on neural networks with shared weights trained with the backpropagation algorithm. They give ratings on a 1-100 scale in a similar fashion to a human using the list interface. In normal backpropagation networks, each connection into a node has its weight which is modified by the backpropagation training. The use of shared weights allows the rating of sequences of variable length, which would be a very hard problem using standard neural network topologies.

In [4]’s paper on Genetic Programming with User-Driven Selection: Experiments on the Evolution of Algorithms for Image Enhancement [4] present an approach to the interactive development of programs for image enhancement with Genetic Programming (GP) based on pseudocolour transformations. In our approach, the user drives GP by deciding which individual should be the winner in tournament selection. The presence of the user does not only allow running GP without a fitness function but it also transforms GP into a very efficient search procedure capable of producing effective solutions to real-life problems in only hundreds of evaluations.[4] used an approach where 3 instead of asking the user to assign a numerical fitness to all the individuals in a population or to directly select the ones to be used to create the next generation (the strategy used in most of the papers described in the previous section), we ask the user to influence tournament selection by interactively comparing pairs of solutions and determining the winner.

Paper [5] proposed a genetic algorithm-based model of a novice jazz musician learning to improvise. GenJam maintains hierarchically related populations of melodic ideas that are mapped to specific notes through scales suggested by the chord progression being played. As GenJam plays its solos over the accompaniment of a standard rhythm section, a human mentor gives real-time feedback, which is used to derive fitness values for the individual measures and phrases.

The aim of this [6]’s (Neurogen, Musical Composition Using Genetic Algorithms and Cooperating Neural Networks) paper is to produce a piece of coherent music that resembles that typically found in traditional hymns. A set of neural networks are used to capture the conceptual ideas that build 'good' music and this knowledge is then used to direct a search for the ultimate composition. Genetic algorithms hold many member states representing partial musical fragments. The neural networks cooperate to produce a heuristic value that represents the worth of each of these musical fragments. This value is then used to evolve better compositions based on fragments with high fitness values. The use of Neural Networks as an evaluation function has proved successful in the guidance of genetic algorithms.

Paper [7] is 1999 ("Sample MIDI files") In this paper, the problem of identifying the melodic track of a MIDI file in imbalanced scenarios is addressed. A polyphonic MIDI file is a digital score that consists of a set of tracks where usually only one of them contains the melody and the remaining tracks hold the accompaniment. This leads to a two-class imbalance problem that, unlike in previous work, is managed by over-sampling the melody class (the minority one) or by under-sampling the accompaniment class (the majority one) until both classes are the same size.

In paper[8], The GTZAN dataset's songs were categorised into seven genres and used in [8]. The stereo channels were then combined into one mono channel, and the music data was converted into a spectrogram using the SoX (SoundeXchange) command-line music application utility, which was then sliced into 128x128 pixel images, and the labelled spectrogram was used as inputs to the dataset, which was split into 70% training data, 20% validation data, and 10% test data. All of the weights were now initialised using the Xavier initialization method. The first four layers are convolutional layers with a kernel size of 2x2 and a stride of two, followed by a max pooling layer. Following the first four layers is a fully connected layer in which each output of the previous layer is fed into each input of the fully connected layer. This yields a vector of 1024 numbers. After that, a SoftMax layer is applied to generate seven outputs, one for each genre. The CNN implementation appeared to be overfit, as the accuracy for the training data was 97% versus 47% for the test data.

In paper [9], They used to train the system by categorizing the music database into different genres. After that, each song must go through a pre-processing stage. Feature Vector Extraction is performed in Python using the librosa package, also known as MFCC.The Mel Scale Filtering is then performed to obtain the Mel Frequency Spectrum by [9]. They obtained two types of feature vectors: Mel Spectrum with 128 coefficients and MFCC coefficients. ConvNet architectures are built using three types of layers: Convolutional Layer, Pooling Layer,and Fully Connected Layer. The database thus obtained is the MFCC, with a genre array size of 10 arrays. The input consists of 1000 songs with ten labels. The vector of features for MFCC The Anaconda Python package was used for the evaluation. The learning accuracy of the Mel Spec feature vector and the MFCC feature vector was found to be 76% and 47%, respectively.

In paper [10] the stages of the proposed method were used in their research. Data collection and selection, pre-processing, feature extraction and selection, classification, evaluation, and measurement are the following methodologies used in [10]. This research makes use of the Spotify music dataset, which contains 228,159 songs across 26 genres and 18 features. To process data structures and perform data analysis, the Python programming language is used, along with the Python Data Analysis Library (PANDAS). In addition, Scikit Learn is used in this study, which is a package containing important modules for machine learning projects. Later on, this genre feature will be used to classify the target. This is done especially for the learning process that uses the SVM classifier. Using hyperparameter, the SVM-RBF classification was carried out by searching the grid for the best results. Kfold cross-validation with free random conditions and a comparison of 80% of training data and 20% of test data. The accuracy for the SVM, KNN, and NB was 80%, 77.18%,and 76.08%, respectively, in [10].

In the paper [11], Mingwen Dong states that they used deep learning, particularly convolutional networks(CNNs), which has lately been used successfully in computer vision and speech recognition. In the process of Musical Information Retrieval (MIR) one specific example is music genre classification. MFCC (Mel-frequency cepstral coefficients), texture, beat, and other human-engineered features have traditionally yielded 61% accuracy in the 10-genre classification task. They used a "divide-and-conquer" method to solve the problem: they split the spectrogram of the music signal into consecutive 3-second segments, made predictions for each segment, and then combined the predictions. [11] used the mel-spectrogram as the input to the CNN to further reduce the dimension of the spectrogram. For Prediction and Training 1000s of music tracks (converted to Mel-Spectrogram) are evenly divided into training, validation, and testing sets in a 5:2:3 ratio. During testing, all music (Mel-Spectrogram) is divided into 3-second segments with 50% overlap. The trained neural network then predicts the probabilities of each genre for each segment in [11]. Their model is the first to achieve human-level 70% accuracy in the 10-genre classification.

In [12] titled Genetic Algorithms in Music Composition: A Comprehensive Review- Focuses specifically on the use of genetic algorithms in music composition, this survey provides a comprehensive review of existing literature. It covers different aspects, such as user interaction, algorithmic control, and the incorporation of fitness criteria in the context of synthesizer design. The survey aims to identify trends, challenges, and emerging themes in the intersection of genetic algorithms and music synthesis.

Paper [13] titled Automated Melody Generation: A Genetic Algorithm Approach in Music Synthesis - concentrates on the role of genetic algorithms in automating the generation of melodies within the realm of music synthesis. It discusses how genetic algorithms have been employed to evolve melodies based on user-defined criteria, highlighting the potential for creating aesthetically pleasing and unique musical compositions. The survey also addresses challenges and opportunities in this specific application of genetic algorithms.

In paper [14] about User-Driven Evolutionary Music Synthesis: A Survey of Interactive Approaches- Focuses on user-driven aspects, this survey explores how genetic algorithms in music synthesis have been designed to incorporate user preferences and interactions. It discusses systems that allow users to define melody types, provide input through simple instructions, and evaluate compositions based on project-specific fitness criteria. The survey aims to provide a comprehensive understanding of the user-centric applications of genetic algorithms in music synthesis.

[15] elaborates upon Genetic Algorithms in Real-Time Music Composition- This investigates the application of genetic algorithms in real-time music composition, going beyond conventional synthesis methods. It explores how genetic algorithms contribute to creating music dynamically, allowing for on-the-fly adjustments and improvisations. The survey aims to uncover advancements, challenges, and potential future directions in the integration of genetic algorithms for real-time music synthesis.

In summary, the cited research papers explore diverse applications of genetic algorithms, genetic programming, and deep learning in the domain of music composition, MIDI file analysis, image enhancement, and music genre classification. These approaches leverage interactive evolutionary computation, neural networks, and convolutional networks to generate music, analyze MIDI files, enhance images, and classify music genres. The studies showcase the versatility of these computational techniques in solving complex problems and underscore the importance of user interaction in shaping evolutionary processes. Additionally, issues such as overfitting in CNN implementations and the varying accuracy of different feature vectors in music genre classification are highlighted, providing insights into the challenges and successes of applying evolutionary and deep learning methods in music-related domains.

# METHODOLOGY

The primary goal is to enhance the music composition process through the integration of a genetic algorithm. The proposed methodology also follows a systematic workflow, involving a deep understanding of genetic algorithms, their implementation in music synthesis, and an evaluation of their impact on the quality and uniqueness of generated musical compositions.

## Workflow

1. Selection of Melody Type:

Users have the option to choose the type of melody they want to generate, whether it's a single-note melody or a stack of notes referred to as chords.

1. Note Instructions:

Users can provide simple mathematical instructions using understandable semantic labeling. These instructions act as a determining factor in how melodies are generated by the system.

1. Melody Generations:

The system generates a finite number of musical melodies based on the provided note instructions. This step serves as part of the modeling process.

1. Fitness of Computer Generations:

Users are involved in the evaluation process by choosing how well a musical generation fits their personal or business project. Each generation is assigned an integer score. The top-scoring generations are then analyzed by the model for further steps.

1. Reproductions:

The model regenerates melodies from the top-scoring generations using simple cut and combination operations. The top two hits of this process are presented to the user.

1. Generation of MIDI Files:

Once the user decides on the final generations, the system generates MIDI files coresponding to those melodies. These MIDI files are stored in the local folder.

1. Downloading MIDI Files:

Users can download the generated MIDI files. This feature provides flexibility for users to use the synthesized music on different Music Digital Audio Workstations(DAWs)

## Block Diagram

A diagram of a variety of blue squares

Description automatically generated

## Algorithmic Implementation

Genetic Algorithms are a variety of complicated, adaptive algorithm that is commonly used to solve issues involving robust efficiency.

## Objective of the Algorithm:

The primary goal of the algorithm is to generate short compositions, specifically targeting compositions with four 4/4 measures.

## Representation of Compositions:

Compositions are represented by a single array of integers that contains both pitch and duration information. This representation allows the algorithm to work with musical elements in a structured manner.

## Input Parameters:

Various input parameters govern the algorithm, including the duration of the composition, tonality, the number and range of tones allowed, the number of iterations, completion criteria, and the interpretation of algorithm results.

## Composition Quality Assessment:

The quality of compositions is assessed based on several criteria, including the composition's similarity to a baseline or reference, values of intervals, sets of "good" and "bad" tones, allowed deviation from reference values, and weight factors influencing different assessment criteria.

## Establishing Assessment Criteria:

The text emphasizes the importance of defining criteria to assess composition quality. These criteria include factors such as the time between subsequent tones, divergence from reference values, and the quantity of "poor" tones.

## Genetic Algorithm Principles:

The algorithm employs principles of Genetic Algorithms to scan the composition search space and identify compositions that are deemed "good enough." This process starts with a random selection of compositions, and the algorithm iteratively refines them using GA operators such as crossover and mutation.

## Fitness Calculation:

Fitness values are calculated for each individual in the population during each iteration. The fitness is inversely related to the quality of the composition; as the fitness drops, the composition is considered "better."

## Selection and Output:

New individuals are formed through mutations of the best ones, and a selection is made among all new individuals and those from the preceding generation. The output of the algorithm is a composition considered ideal based on the established parameters and iteration procedure.

CODE COMPOSITION

*Genetic Algorithm*:

This part of the code defines the genetic algorithm components such as genomes, populations, crossover, mutation, and other utility functions. The genetic algorithm aims to evolve a population of genomes to meet a specific fitness criterion.

*Functions and Types:*

*Genome:* Represents an individual in the population, which is a list of integers (0 or 1).

Population: A list of genomes.

*PopulateFunc*: A function that generates an initial population.

*FitnessFunc:* A function that evaluates the fitness of a genome.

*SelectionFunc:* A function that selects genomes from the population for reproduction.

*CrossoverFunc:* A function that performs crossover (mating) between two parent genomes.

*MutationFunc*: A function that applies mutations to a genome.

*PrinterFunc*: A function to print statistics about the population.

*Key Functions:*

*generate\_genome*(length: int) -> Genome: Generates a random genome of a specified length.

*generate\_population*(size: int, genome\_length: int) -> Population: Generates an initial population.

*single\_point\_crossover*(a: Genome, b: Genome) -> Tuple[Genome, Genome]: Performs single-point crossover between two genomes.

*mutation*(genome: Genome, num: int = 1, probability: float = 0.5) -> Genome: Applies mutations to a genome.

*population\_fitness*(population: Population, *fitness\_func: FitnessFunc*) -> int: Calculates the total fitness of a population.

*selection\_pair*(population: Population, fitness\_func: *FitnessFunc)* -> Population: Selects a pair of genomes for reproduction.

*run\_evolution*(...) : Main function to run the genetic algorithm.

MIDI Music Generation:

This part of the code is responsible for converting genomes generated by the genetic algorithm into MIDI music.

*Functions:*

*int\_from\_bits*(bits: List[int]) -> int: Converts a list of bits to an integer.

*genome\_to\_melody*(...) -> Dict[str, list]: Converts a genome to a musical melody.

*genome\_to\_events*(...) -> List[Events]: Converts a genome to a series of musical events.

*fitness*(...) -> int: Evaluates the fitness of a genome by playing the music using the Pyo library.

*metronome*(bpm: int): Defines a metronome sound using Pyo.

*save\_genome\_to\_midi*(...): Saves a genome's melody to a MIDI file.

*Librosa and Matplotlib Usage:*

*Librosa* is used for audio processing, including loading audio, computing STFT, and converting to a spectrogram.

*Matplotlib* is employed for creating spectrogram plots with adjustable parameters such as hop length and axis scaling.

The code provides insights into transforming audio data into a visual representation, aiding in the analysis of frequency content over time.

*Audio Spectrogram Visualization:*

This code segment focuses on visualizing the spectrogram of an audio file using Librosa and Matplotlib.

It loads an audio file, computes the Short-Time Fourier Transform (STFT), and calculates the magnitude spectrogram.

Two spectrogram plots are generated: one in linear scale and another in logarithmic scale for better visualization of dynamic range.

# EXPERIMENT

Here is the visualization and comparison of both linear and logarithmic spectrogram plots. The adoption of a logarithmic scale is particularly advantageous for enhancing the visualization of the dynamic range within audio signals. This proves especially beneficial in scenarios where audio exhibits a mix of quiet and loud components, allowing for a more effective representation of the overall signal characteristics.

*Audio 1:*

Parameters: num\_bars=8, num\_notes=4, num\_steps=1, pauses=True, key="C", scale="lydian", root=4,

population\_size=10, num\_mutations=2, mutation\_probability=0.5, bpm=128, rating=2

A graph of a sound wave

Description automatically generated

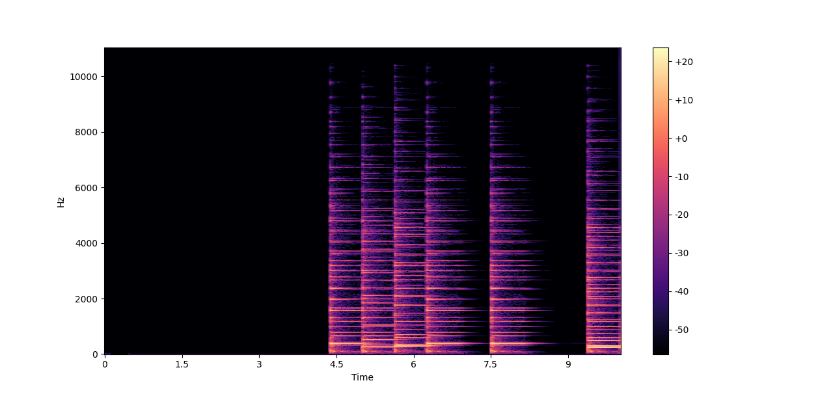
 FIG 1-Visual Image

FIG 2 - Linear Spectrogram

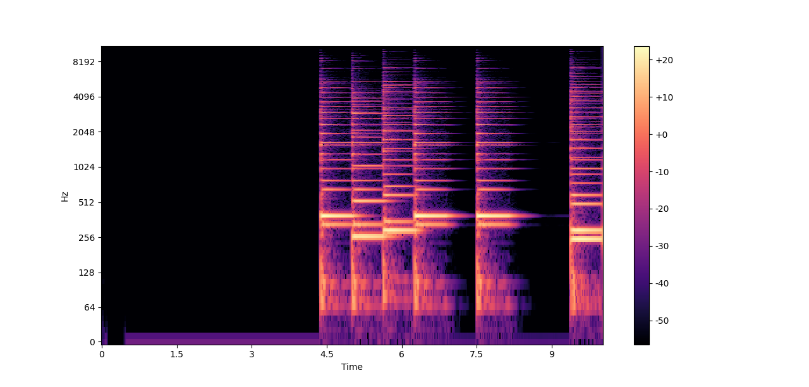


FIG 3 - Logarithmic Spectrogram

*Audio 2:*

Parameters: num\_bars=8, num\_notes=4, num\_steps=1, pauses=True, key="C", scale="lydian", root=4,

population\_size=10, num\_mutations=2, mutation\_probability=0.5, bpm=128, rating=1

A graph of a graph showing a number of blue lines

Description automatically generated with medium confidence

FIG 4 - Visual Image

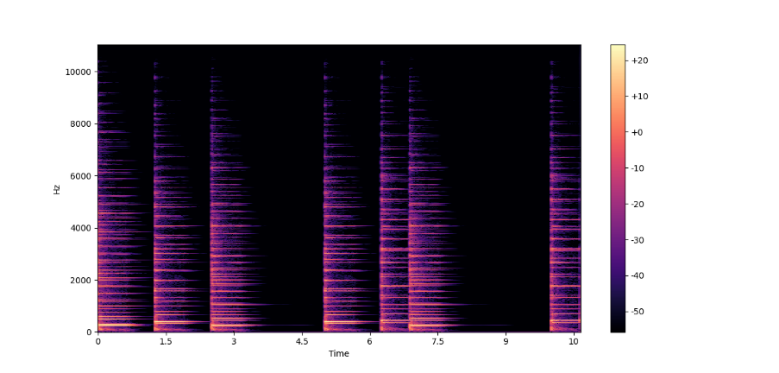


FIG 5 - Linear Spectrogram

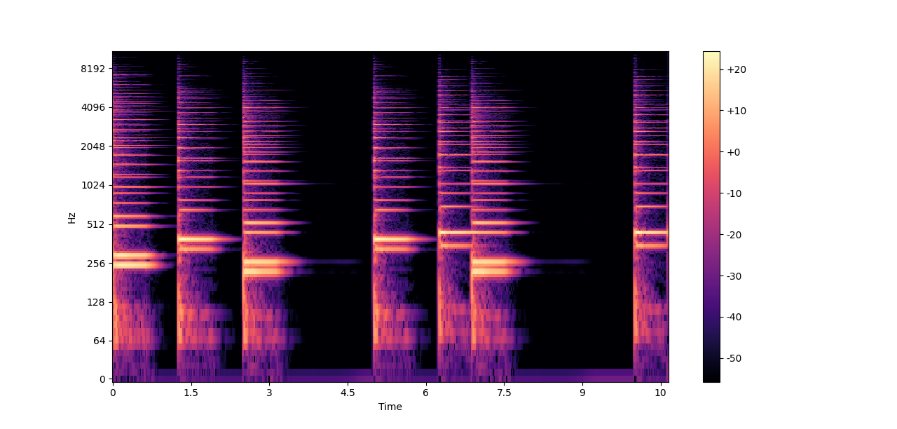


FIG 6 - Logarithmic Spectrogram

*Audio 3:*

Parameters: num\_bars=8, num\_notes=4, num\_steps=1, pauses=True, key="C", scale="lydian", root=4,

population\_size=10, num\_mutations=2, mutation\_probability=0.5, bpm=128, rating=5

A graph of blue lines

Description automatically generated

FIG 7 -Visual Image

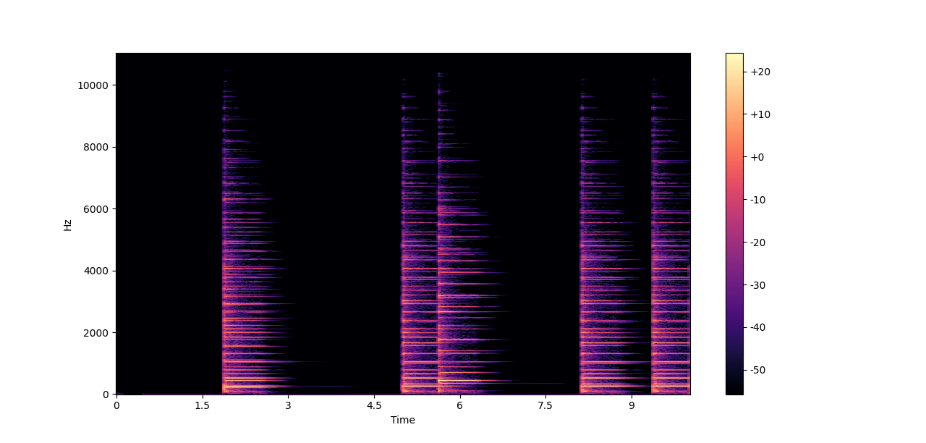


FIG 8 -Linear Spectrogram

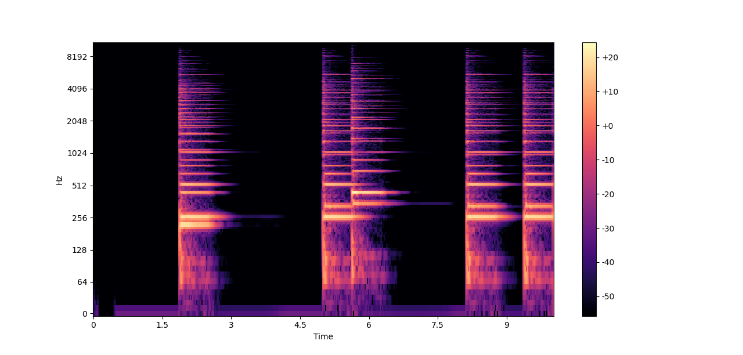


FIG 9 -Logarithmic Spectrogram

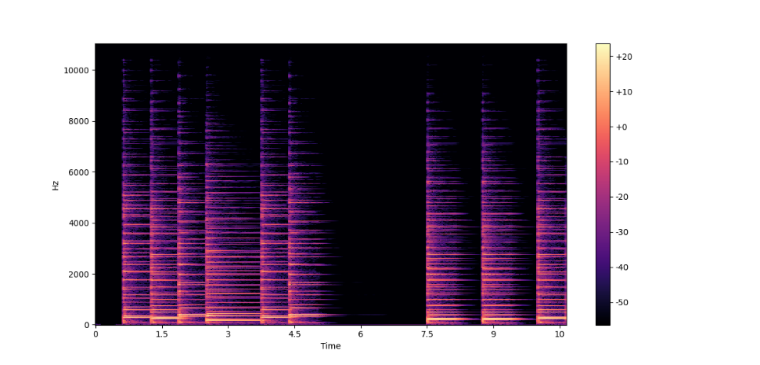
*Audio 4:*

Parameters: num\_bars=8, num\_notes=4, num\_steps=1, pauses=True, key="C", scale="lydian", root=4,

population\_size=10, num\_mutations=2, mutation\_probability=0.5, bpm=128, rating=4

A graph of a sound wave

Description automatically generatedFIG 10 -Visual Image



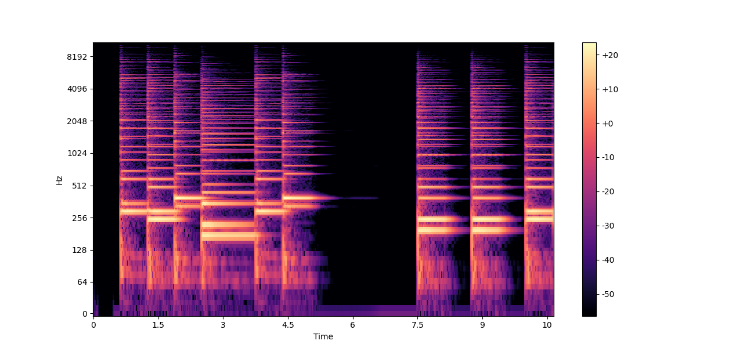
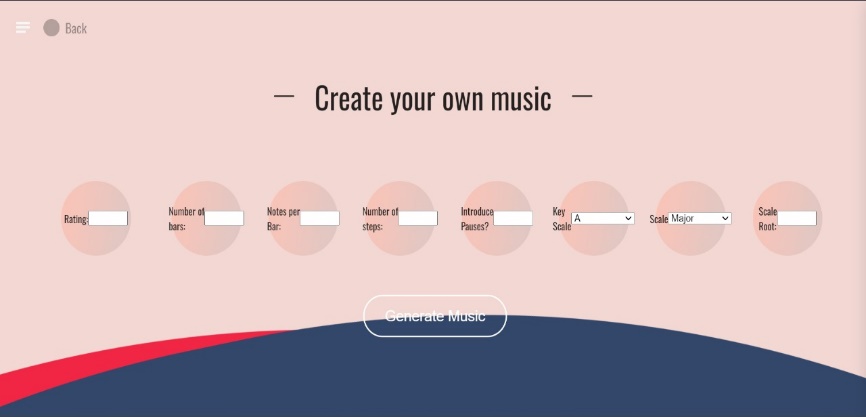
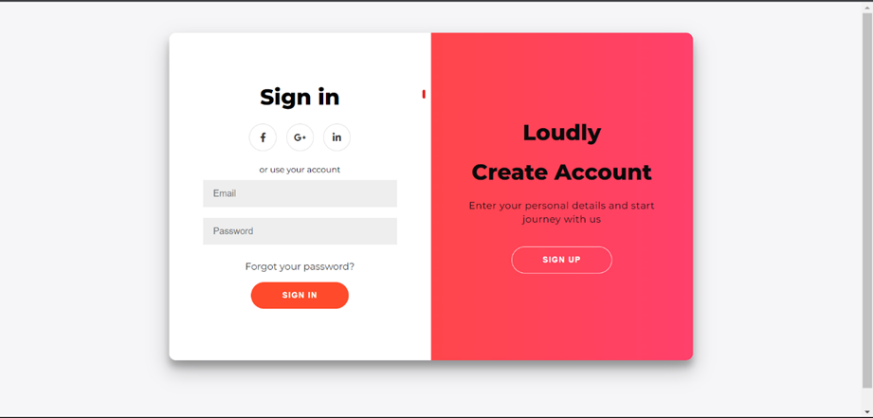
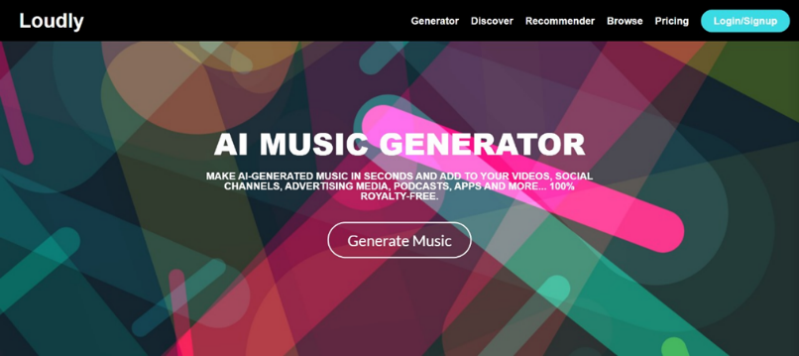
 FIG 11 -Linear Spectrogram

FIG 12 -Logarithmic Spectrogram

# RESULTS AND DISCUSSION

Incorporating a genetic algorithm into the music composition technique has yielded significant success in enhancing the music-writing process. Offering choices between single-note and chord-based melodies, the system empowered users to exercise their preferences, showcasing its adaptability to a diverse range of musical tastes influenced by user-supplied mathematical instructions. The algorithm's capacity to replicate and adjust to user input was exemplified through the generation of a varied repertoire of tunes, demonstrating its flexibility and responsiveness to individual creative preferences.

The introduction of a subjective element in the assessment process during the fitness evaluation, where users actively assigned scores to the generated melodies, added a nuanced layer to the evaluation framework. Subsequently, the algorithm reproduced the highest-scoring generations through cut and combine procedures, effectively creating new compositions. This iterative refinement process, exemplified by presenting the top two hits to users, not only demonstrated the algorithm's adaptability but also underscored its ability to continuously improve and tailor musical outputs based on user feedback.



# CONCLUSION

The study introduces a genetic algorithm-based

approach for creating aesthetically pleasing musical compositions with characteristics such as pleasing intervals and understandable rhythm. This innovative method allows effective regulation of composition quality, utilizing reference persons and pre-defined specifications to shape rhythmic and harmonic elements. Coding techniques play a crucial role in providing rapid control over composition, tones, and rhythm, utilizing arrays and mathematical functions. The study envisions potential expansions, suggesting the exploration of additional metaheuristics for comparison or hybridization with the genetic algorithm. Furthermore, there is interest in adapting the genetic algorithm to generate compositions spanning various music genres through parameter alterations. In essence, the study contributes a versatile and controlled framework for musical composition, inviting further exploration into its applications and potential enhancements.

# REFERENCES

1. D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA: Addison-Wesley Professional, 1989.
2. N. Tokui, H. Iba, “Music Composition with Interactive Evolutionary Computation,” in Proc. Generative Art2000, the 3rd International Conference on Generative Art, Milan, Italy, 2000.
3. Johanson, B. E. (1997) "The GP-Music System: Interactive Genetic Programming for Music Composition", University of Birmingham, Second-Semester Mini-Project Report.
4. Genetic Algorithms in Music Composition
5. Automated Melody Generation- A Genetic Algorithm Approach in Music Synthesis
6. Poli, R., Cagnoni, S., “Genetic Programming with UserDriven Selection: Experiments on the Evolution of Algorithms for Image Enhancement,” Genetic Programming 1997: Proceedings of the Second Annual Conference, Morgan Kaufmann, 1997.
7. Biles, J. A. (1994). GenJam: A genetic algorithm for generating jazz solos. In Proceedings of the 1994 International Computer Music Conference, ICMA, San Francisco.

[8] Gibson, P. M., Byrne, J. A., “Neurogen, Musical

Composition Using Genetic Algorithms and

Cooperating Neural Networks,” IEE Conference

Publication, No. 349, pp. 309-313, 1991.

[9] Rosa, A.C.(1999)"Sample MIDI files", http://

www.geocities.com/ResearchTriangle/Station/l2

32/ geamusic.html

[10]Mingwen Dong deep learning, particularly convolutional networks(CNNs)

[11]User-Driven Evolutionary Music Synthesis: A Survey of Interactive Approaches

[12] Feng, B., Jiang, Z., Fan, Z., Fu, N. (2010). "A Method for Member Selection of Cross-Functional Teams Using the Individual and Collaborative Performances."

[13] Pelchat, C., Craig M. Conducted research on the GTZAN dataset, categorizing songs into seven genres and using convolutional neural networks for classification.

[14] Genetic Algorithms in Real-Time Music Composition

[15] Meenakshi, K. Conducted a survey using ConvNet architectures for music genre classification, using features extracted with MFCC.