A decorative graphic on the left side of the slide consisting of two overlapping parallelograms. The front one is blue and the back one is light green. They are positioned diagonally, with the blue one partially covering the green one.

# An Evolutionary Algorithm to Evolve Music

Austin Lee

# A Quick Introduction

- I created an evolutionary algorithm to evolve music
- Algorithm Overview:
  - Generates an initial population of random music compositions
  - Genes based on chromatic scale between C4 and C5
  - Mutates genome and calculates fitness
  - Fitness is based on fitness function I made and user input via interactive evolution
  - Utilizes simple tournament selection to remove least fit individuals
  - Returns top performing compositions after X generations

A Chromatic Scale starting at C4



# The genome

- 4 measures, 16 beats
- Notes ordered from 0-13
  - 0 is a rest
  - 1 corresponded to C4, 2 to C#
  - Increased by half steps

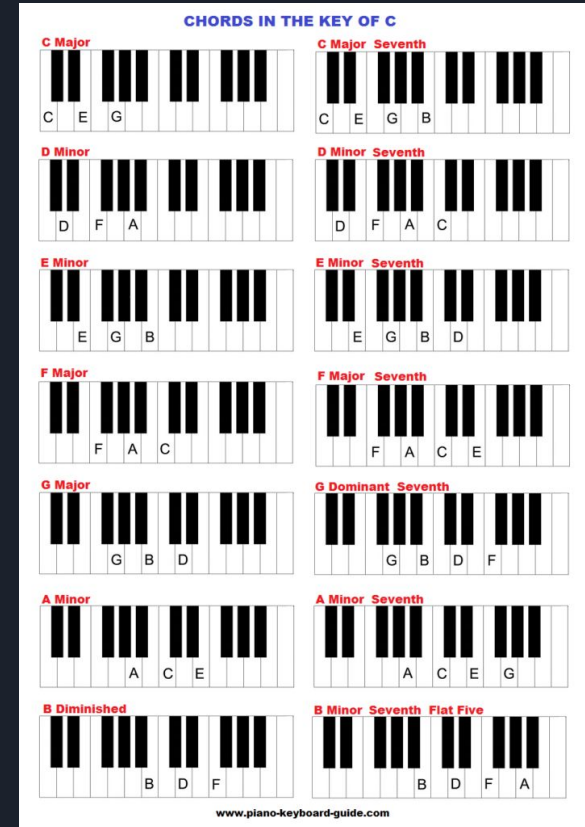
An example of a randomly generated initial parent

[ 5 9 3 2 11 6 4 0 12 2 12 8 9 11 10 0]



# The Fitness Function


- Tested for 3 main criteria
  - Similarity to 14 C major chords
    - Examples are C major (CEG, [1,5,8,13]), D minor (DFA, [3,6,10]), or E minor (EGB, [5,8,12])
    - Calculated hamming distance between a single measure and the chord notes to quantify similarities
  - Repetitions within each measure or between two measures
    - More repetition signifies higher fitness
  - Increasing/decreasing over different steps
    - For example, increasing by 1 half step over 3 notes would probably sound worse than increasing by 2 half steps, or a whole step, over 3 notes.
    - C -> C# -> D vs C -> D -> E



# Interactive Evolution

- In terms of music, my fitness function is super handwavey. Is it possible to quantify how good a composition sounds based on fancy math and algorithms alone? Prob not lol
- Enter interactive evolution
  - Every X generations, the user could pick which compositions sounded the best
  - I found every 10 generations to work well
  - Also lower parents/children
  - The most enjoyed compositions would be awarded a large amount of fitness


Genome Number: 1  
Genome String: [ 5 12 9 6 6 1 1 8 0 6 10 1 12 0 9 10]



0:00 -0:09

=====

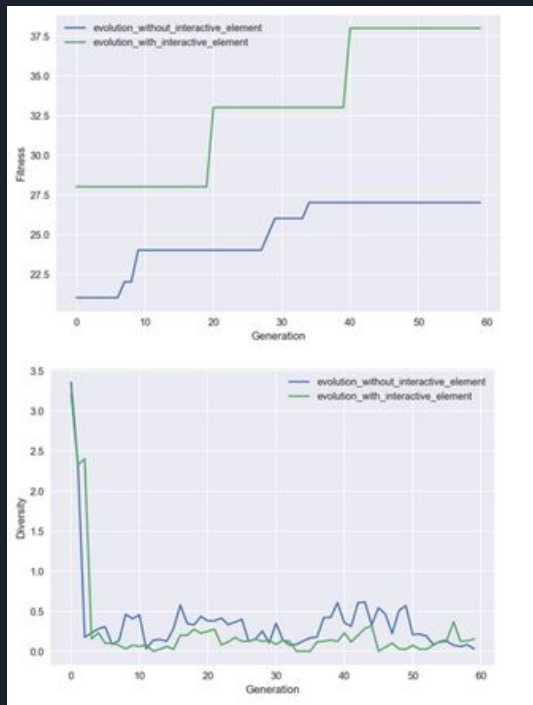
Genome Number: 2  
Genome String: [ 8 11 9 8 9 3 0 7 1 3 12 4 2 1 5 7]



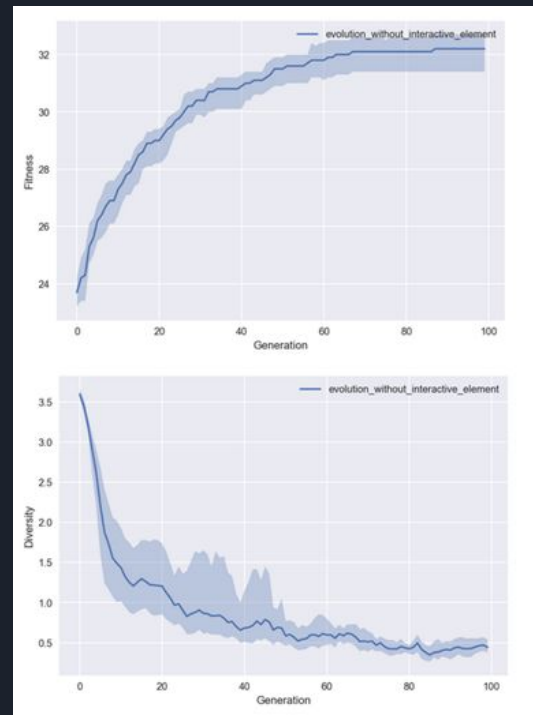
0:00 -0:09

The image displays two examples of music generated from genome strings. Each example includes the genome number, the genome string (a list of integers), a musical notation staff, and a playback control bar. The first example is for Genome Number 1 with a genome string of [ 5 12 9 6 6 1 1 8 0 6 10 1 12 0 9 10]. The second example is for Genome Number 2 with a genome string of [ 8 11 9 8 9 3 0 7 1 3 12 4 2 1 5 7]. Both musical notations are in 4/4 time and use a treble clef. The playback control bars show a duration of 0:00 to -0:09, indicating a 9-second clip.

# Results



Fitness and Diversity with and without the interactive evolution. 1 run, 5 parents/children, 60 generations.



Fitness and Diversity over time without interactive evolution. 10 runs, 50 parents/children, 100 generations.



Without interactive evolution ->

- More repetition, less half steps

Final Evolution, Best Solution

Worst Solution

With interactive evolution ->

- Some repetition, more half steps

Final Evolution, Best Solution

Worst Solution



# Live Demo

I could not figure out how to export mp3s of my compositions, so I have the live demo. Doesn't feature the written composition part since music21 was weird with mac....

enjoy!



# Thanks! Any questions?



Me getting ready for piano class (circa 2021, colored)