Evolving ambience using DE

Andrea Barbieri

Data Science and Scientific computing - University of Trieste
Global and Multi-Objective Optimization Final Project

A.Y. 2022/2023

Outline (a path is formed by laying one stone at a time)

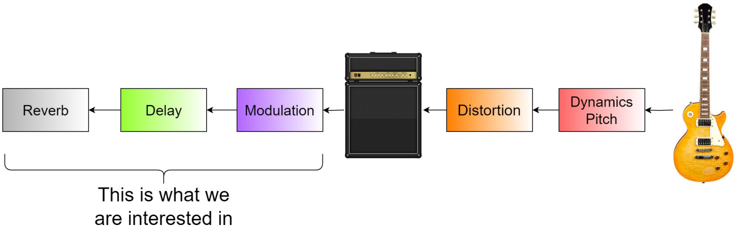
Introduction

Introduction

- The aim of the project is to use DE to set the parameters of a guitar pedalboard in order to match, as close as possible, a target sound.
- To define the problem, we first have to take a brief overview of the tools used.

Guitar effects

- Guitar effects are electronic devices used to modify the sound of musical instruments, specifically guitars.
- Sound can be altered in a variety of ways and five main families can be recognized:



Plugins used

- For this project three freeware plugins were used, all developed and distributed by Valhalla DSP:
    - **Space Modulator**: modulation (chorus/flanger/delay), 5 parameters, 11 modes
    - **Freq Echo**: delay, 6 parameters
    - **Supermassive**: delay/reverb, 9 parameters, 18 modes
- Space Modulator and Supermassive have several modes that change the internal algorithm and how the sound is treated, resulting in very different effects.

Spectrograms

- To check similarity between two sounds **spectrograms** have been used: these are visual representations of audio sources in both the frequency and time domain.
- Since both the target audio and the results of the algorithm use the same "dry" signal, one could easily check for differences by subtracting the two spectrograms.

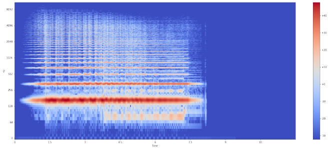


Figure: No effects.

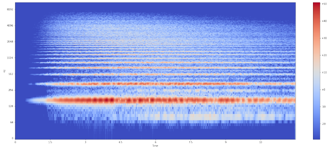


Figure: Reverb applied.

Towards problem formalization

- To adapt the problem to the EA framework, each individual will represent a **combination of all the parameters**.
- Each time we have to assess the fitness of an individual, its parameters will be applied to a **common pedalboard** and a spectrogram will be obtained.
- The fitness function, inspired by a previous work[1] in analogous settings, will be the **Euclidean norm** of the **difference** between the individual's spectrogram and the target one.

[1] Cited paper

The setting

- **Individuals**: vectors of 20 real-valued (scaled in $[0, 1]$) parameters plus 2 categorical ones
- **Fitness function**: Euclidean norm of difference between individual's spectrogram and target
- **Algorithm**:?

JADE
(and variations on the theme)

About JADE

- **JADE** is a major variation of the differential evolution algorithm, that implements the following modifications:
  - A mutation strategy called **"current-to-pbest"**
  - An **external archive** of suboptimal solutions
  - **Adaptive update** of mutation rate and factor

current-to-pbest strategy

- It is a generalization of the *current-to-best* strategy, developed in order to balance exploitation (of the best solutions) and exploration.
- The mutation vector v_i of a given individual x_i is computed as:

$$v_i = x_i + F_i(x_{best}^p - x_i) + F_i(x_{r1} - x_{r2}),$$

where x_{best}^p is a sampled individual from the top-p performing ones, x_{r1}, x_{r2} are randomly sampled individuals in the population and F_i is the usual mutation factor.

External archive (I)

- JADE implements an **archive** to maintain past solutions that have not passed the selection but may still be useful to explore the search space.
- The archive is initialized as empty and as parents fail selection, they are added to the archive until the max size is reached.
- Once the archive is full, new individuals are added randomly overwriting past individuals.

External archive (II)

- The archive is used during the creation of the mutation vector as follows:

$$v_i = x_i + F_i(x^p_{best} - x_i) + F_i(x_{r1} - \tilde{x}_{r2}),$$

where $\tilde{x}_{r2}$ is an individual sampled from the union between current population and archive.

Adaptive hyperparameters (I)

- The **crossover probability** and **mutation factor** are not fixed as in traditional DE, but are **randomly generated** in each generation.
- Furthermore, the distribution used to generate the values takes into account successful past mutation, adapting accordingly.

Adaptive hyperparameters (II)

- The crossover mutation and mutation factor of each individual is randomly generated from

$$p_i \sim N(\mu_{CR}, 0.1),\ F_i \sim \text{Cauchy}(\mu_F, 0.1).$$

- μ_{CR} and μ_F are adaptive parameters initialized to 0.5 and updated as

$$\mu_{CR} \leftarrow (1-c)\mu_{CR} + c\ \text{mean}(S_{CR}),$$
$$\mu_F \leftarrow (1-c)\mu_F + c\ \text{mean}_L(S_F)$$

where c is a user-defined parameter, S_{CR} and S_F are the sets of crossover probabilities and mutation factors of the successful individuals in the previous generation.

Categorical parameters

- A crucial point is how to deal with **categorical parameters**, which are not usually found in the DE framework.
- When computing the donor vector, the mode of the effects is randomly inherited from one of the donors (the p-best individual and the two random ones) with uniform probability.
- Uniform crossover then applies as usual.

A Lamarckian take on JADE

- After the first experiments, a **stagnation** problem was found in the population, prematurely converging to a local optimum.
- To overcome this problem and inspired by related works[2], JADE was combined with local search methods, in particular a modified version of the **Hooke-Jeeves algorithm**.

[2] Cited paper

The Hooke-Jeeves algorithm (I)

- The algorithm used is divided in two parts:
  - *Exploratory move*: each gene is randomly perturbed adding a random quantity between $[-\alpha, \alpha]$. If fitness has improved go to the next move, else repeat the process using $[-\alpha/2, \alpha/2]$. The exploratory move iterates until a max number of tries is reached.
  - *Pattern move*: Let x_{base} be the initial genome to optimize and x_{exp} the genome found by the exploratory move. The pattern move consists of replicating this successful mutation returning a genome x_{patt} defined as:

$$x_{patt} = x_{exp} + \beta(x_{exp} - x_{base}),$$

where $\beta > 0$.

The Hooke-Jeeves algorithm (II)

- The HJ algorithm is not used on every single individual, but following specific criteria.
- Let G be the total number of generations and g be the current one. If an offspring has a lower fitness compared to the parent's one, we apply the HJ algorithm if

$$\mathrm{rand}(0,1) < g/G.$$

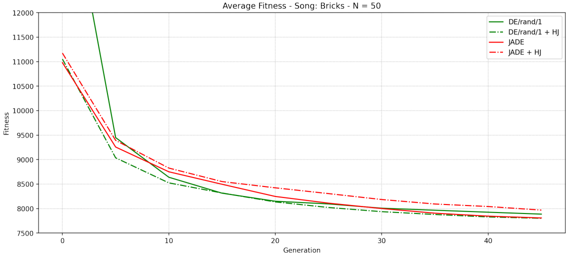
- In other words, the optimization algorithm is more likely to be used in later generations, preserving the population diversity in the first stages and introducing some exploitation mechanisms later on.

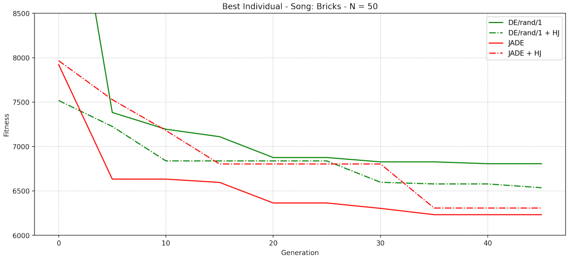
Results and Conclusions

Algorithm evaluation

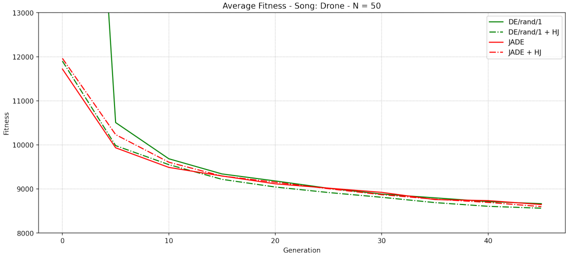
- The algorithm was tested using three different short audio samples, recorded with the same VSTs optimized by the algorithm.
- For comparison, also a plain DE/rand/1 algorithm was used, both with and without the HJ optimization step.
- Each algorithm was run using a population of 50 individuals, evolved for 45 generations. Presented results are the mean values of 3 different runs.

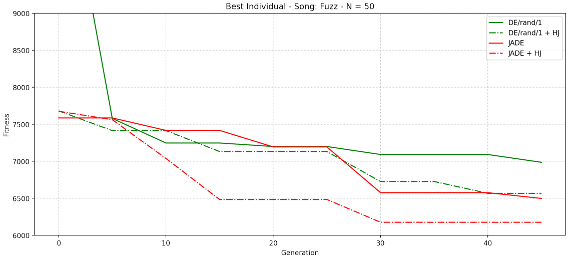
A song about bricks and walls (delay)



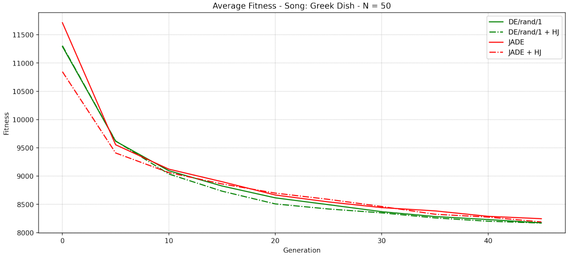


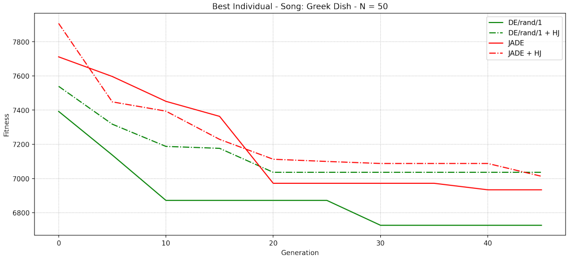
A cinematic droning sound (reverb)





A song about a Greek dish (all)





Qualitative Assessment

- Qualitatively speaking, on the first and second clip the algorithm does a good job in capturing the delay pattern /reverb of the target.
- On the third one, the algorithm barely recognizes the presence of a lush reverb in the background, neglecting delay patterns and chorus effect.

Summing up

- Looking at the experiments, we can clearly see that the population easily gets stuck in a local optima, making the average fitness converging after ~ 20 generations.
- The HJ optimization mechanism does not seem to work in preventing premature convergence.
- Generally, the algorithm seems to work better with single plugins.

Future works

- **Fitness**: using different norms or even different measures, such as spectral centroids.
- **Parallelization**: a possible solution could be the use of an island model with a k-star topology, in order to improve the overall performance of the algorithm and preserve diversity.
- **Coevolution**: a 3-population cooperative approach could be used, restricting each population to just one plugin.

Thank you for the attention!

And thanks for all the fish!