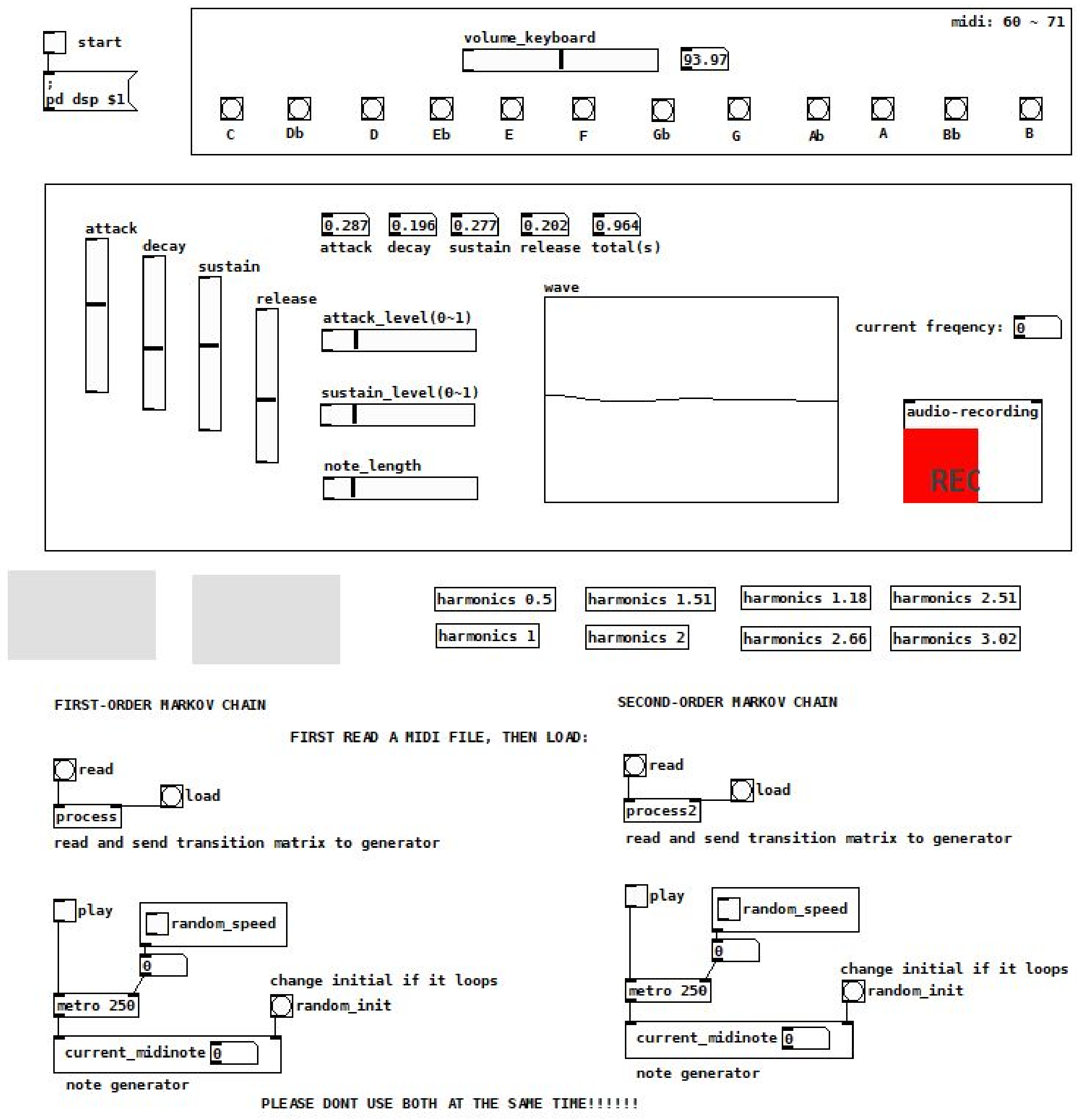
Markov Chain

Tiange Zhu



This project is built based on synthesizer from the previous homework. I implemented both first-order Markov Chain and second-order Markov Chain processes for prediction.

User Guide

The user could choose either first-order Markov Chain or the second-order Markov Chain for prediction. In order to use, first click the bang of [read], select a midi file as reference of prediction in the folder, then click the bang of [load]. It usually takes 10 seconds to store. When finished, click the toggle of [play]. If it is playing a loop of one note, click [random_init] so the system would randomize the initial pitch. The [current_midinote] shows the midi number of current generated note. It is possible to play the generated notes with a randomized speed if the user clicks the toggle of [random_speed].

Synthesizer

The synthesizer was mainly made of three parts: envelope modifier, midi keyboard control, additive harmonics. The envelope modifier is for adjusting the soundwave of each note, additive harmonics patches are for making a harmonic sound. The keyboard control is kept to play the synthesizer real-time, but it wouldn't be necessary for using Markov Chain to auto-generate new pitches. It is also possible to record with audio-recording patch.

First-order Markov chain

In the context of first-order Markov chain, prediction of the next note is based on the current note and a list of probabilities for following notes. In the [process] patch, it first reads a stream of midi notes from the reference, then pack notes into pairs, transform into the form of key+value so it is easier for indexing. Finally, the list of probabilities are stored in a transition matrix through [coll].

When file processing is finished, [process] sends a signal to start [generator]. It randomly generates the next note based on the current note and transition matrix. The frequency of newly generated midi note will be sent to synthesizer.

The speed randomizer controls the speed of note generator to be 200~250ms per note. It automatically changes every 2 seconds.

Second-order Markov chain

Second-order Markov chains predict the next note based on two previous notes and the probability of subsequent notes following those two notes.

In the [process2] patch, it reads and transform the reference into a stream of numbers, the stream is chunked into many midi numbers of three(a pitch and the two pitches after), then transform into key+value form, and store in transition matrix.

[generator2] predicts the pitch of new notes based on its two previous notes and transition matrix. It shows the current midi note and send frequencies to synthesizer to make it sound. The speed randomizer can be triggered, which shares the same setting as the speed randomizer of 1st-order Markov chain. The preciseness of speed could be improved by making a transition matrix of note durations.

I prepared midi file of Bach's “Allemande” as reference. I have initialized parameters, and only showed user control elements, so it's easier to use.