HOW TO USE THE ALGORITHMS

The outline of how the two programs, Sung Melody to Matrix (SMM.m) and Melodic Fidelity Evaluator (MFE.m) work together along with example inputs and outputs is shown in Figure 1.

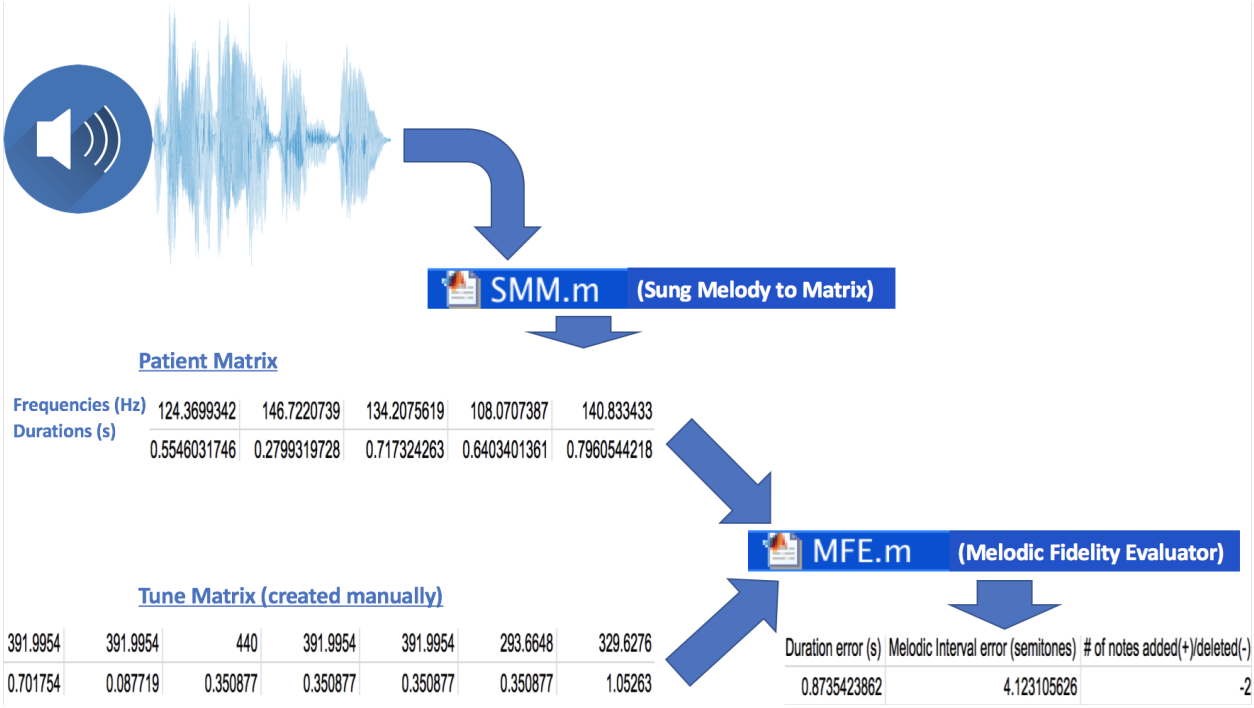
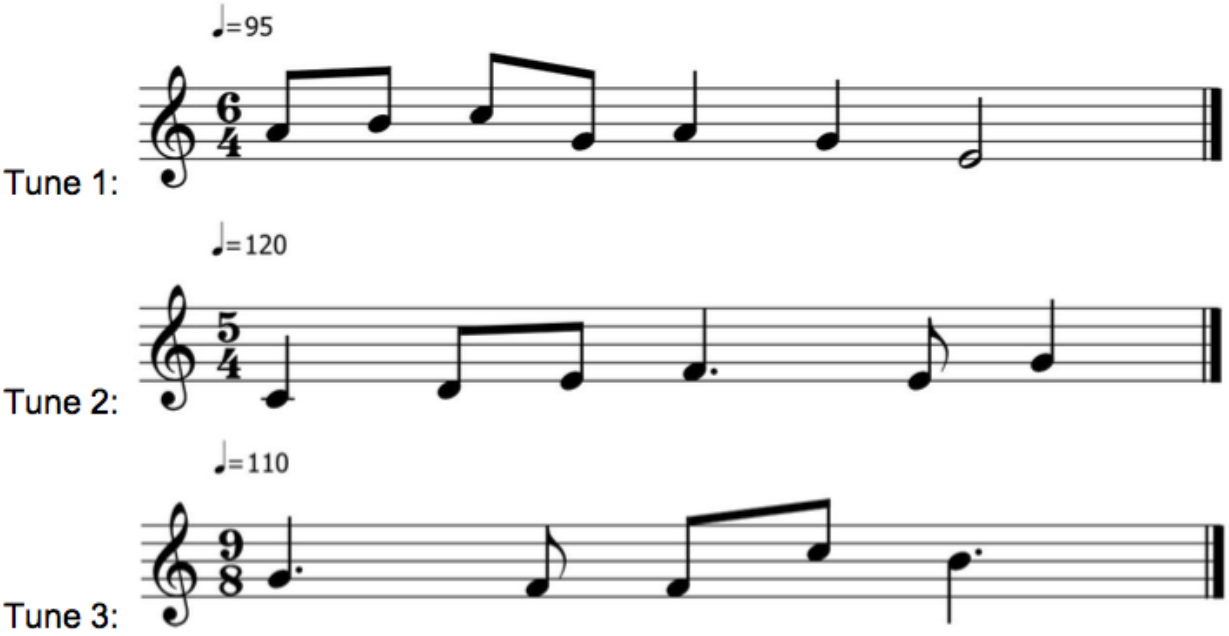


FIGURE 1. The input of the SMM.m is an audio file and the output is a matrix containing the frequencies and corresponding durations for each note sung. The inputs of MFE.m are the output matrix of the SMM.m as well as a matrix of the frequencies and corresponding durations of the notes of the tune the patient was attempting to repeat. The outputs of MFE.m are duration error, note interval error and number of notes added or deleted.

The implementation of the two programs in order to evaluate the repetition ability of a post-stroke patient is as follows:

(1) A simple melody is played to the post-stroke patient. Examples of such melodies are the following:



The WAV files for these tunes can be found here in my GitHub repository, and they are named tune1.wav, tune2.wav and tune3.wav, respectively.

(2) While being recorded, the patient attempts to repeat the played tune.

(3) Open a 2018 version of MATLAB. You will need to have these two toolboxes:
- Image Processing Toolbox (version 10.2 was used)
- Audio System Toolbox (version 1.4 was used)

(4) Clear your workspace variables (using the command clear) and close any figures that may be open (using the command close).

(5) Find out the path of the recording of the patient, which will be put into a variable called filename as a string. An example of this in Mac computers would be:

```
filename='/Users/Androulakis/Recordings/patient_tune1.wav';
```

And likewise in Windows would be:

```
filename='C:\Users\Androulakis\Recordings\patient_tune1.wav';
```

Note that the apostrophe used (') is straight and not curved as in (’). Else, MATLAB will give you an error. Of course, your path does not have to be identical to the one shown above, so this is just an example. This is how your recording will be inputted into SMM.m.

(6) Run SMM.m. The output will be in a variable called PHz (PHz stands for Patient Hertz) containing the frequencies and corresponding durations for each note sung.

(7) Manually create the matrix of the frequencies and durations of the melody the patient attempted to repeat. For finding the frequencies of the notes, Wikipedia's Piano Key Frequencies chart was used. For finding the durations of the notes, use the formula applicable to the tempo of your tune:

which note gets the beat	formula
half note	120/(beats per minute)
quarter note	60/(beats per minute)
eight note	30/(beats per minute)
sixteenth note	15/(beats per minute)

For example, the matrices of the tunes given above in Step 1 would be inputted as follows
(OHz stands for Original Hertz.):
Matrix of Tune 1:

```
OHz=[440 493.8833 523.2511 391.9954 440 391.9954 329.6276;30/95
30/95 30/95 30/95 60/95 60/95 120/95];
```

Matrix of Tune 2:

```
OHz=[261.6256 293.6648 329.6276 349.2282 329.6276 391.9954;60/120
30/120 30/120 90/120 30/120 60/120];
```

Matrix of Tune 3:

```
OHz=[391.9954 349.2282 349.2282 523.2511 493.8833;90/110 30/110
30/110 30/110 90/110];
```

Even though these matrices could be attempted to be computed through the SMM.m, this program works better with the human voice, and thus for better accuracy, manually create the matrices of the original tunes.

(8) Now that you have both the variables PHz and OHz in your MATLAB workspace, run MFE.m. This program will output the Time Error (seconds), Note Interval Error (semitones), and Number of Notes Added(+)/Deleted(-). The time error and note interval error is saved in the variable BR. The number of notes added or deleted is saved in the variable NotesAddedOrDeleted.