Team Project: Octave Band Filtering

ECE 6530: Digital Signal Processing
December 4, 2023

Tyler Bytendorp Miguel Gomez Chase Griswold Benjamin Hayes

University of Utah Electrical and Computer Engineering

Due Date: Dec 5, 2023

5.1 Octave Bands

This bit was taken care of in Matlab and this is the function to set up the filter bands table. The results are below and a copy of the code is included in appendix A: Since we have a limit

val [units]	O_0	O_1	O_2	O_3	O_4	O_5	O_6
Lower (Hz)	65.406	130.81	261.63	523.25	1046.5	2093	4186
Lower (Rad)	0.05137	0.10274	0.20548	0.41096	0.82192	1.6438	3.2877
Upper (Hz)	123.47	246.94	493.88	987.77	1975.5	3951.1	7902.1
Upper (Rad)	0.096974	0.19395	0.3879	0.77579	1.5516	3.1032	6.2063
Center (Hz)	94.439	188.88	377.75	755.51	1511	3022	6044.1
Center (Rad)	0.074172	0.14834	0.29669	0.59338	1.1868	2.3735	4.747

Table 1: Frequency Ranges for Octaves 0 to 6 starting with C_2

on the bands we can recognize that arises from the use of the sampling freq of $8kHz$, we can only obtain unique detection for the set of Octaves whose frequencies are below the Nyquist rate of $\frac{fs}{2}$ or $4kHz$. Or, O_5 in the table above.

5.2 Octave Filter Bank

The band pass filter bank was put together in python and has the added benefit of being more portable than the use of Matlab for this application. These are the results from the creation of the x signal needed to pass on to the filter bank. Defining the expression below:

$$x(t) = \begin{cases} a & t, \\ b & t, \\ c & t \end{cases}$$

and now the code to do this along with the plot showing the data is what we expect:

Appendix A

```
function print_octaves(n,fs)
% n should be a range relative to A4. ex: -1:1 gives A3,A4,A5
A4 = 440;
C4 = A4.*2.^(-9./12);
B4 = A4.*2.^(2./12);
Octaves = 2.^n;
Cs = C4.*Octaves;
Bs = B4.*Octaves;
n_range = 0:length(n)-1;
Centers = (Cs + Bs)./2;
cell(size(Centers));
octave_array = arrayfun(@(x) sprintf('Octave %d', x), n_range, 'UniformOutput',
    false);
w_Cs = Cs.*2.*pi./fs;
w_Bs = Bs.*2.*pi./fs;
w_Centers = Centers.*2.*pi./fs;
rows = {'Lower (Hz)','Lower (Rad)','Upper (Hz)','Upper (Rad)','Center (Hz)','Center
     (Rad)'};
% Summarize data in a table
T = array2table([Cs; w_Cs; Bs; w_Bs; Centers; w_Centers],'VariableNames',
    octave_array,'RowName',rows);
disp(T)
disp('Hz are not normalized.')
disp('Radians are normalized by sampling frequency.')
end
```