ECS730 Digital Audio Effects

Assignment 2: Audio Effect Programming

Instructions:

Implement a multiband dynamic range compressor audio effect. The effect should be built as a VST plug-in using the Juce platform (www.rawmaterialsoftware.com), a C/C++ library which can build industry-standard VST and AudioUnit plug-ins.
You can use the provided code for a dynamic range compressor and for equalization as starting points. The labs should have all the required software, but this can also be done from home.
Instructions on setting up Juce can be found below, and example code templates are available on QMplus.
Where possible, include user-tunable parameters in your effect. The example code includes user interface components for adjusting parameters. You should also customize the interface and comment your code to highlight that it is your own.

Turn In:

Using the online submission system, submit a ZIP archive containing:

• **Audio files** of your effect in action (at least two processed samples), *plus* original version of any source material used.).
• **Commented Code** used to run the effect. Please make your code legible and easy to understand! Please also indicate how your code should be compiled, especially if you use tools other than the provided Juce library.
• **PDF Report**, 3 or 4 sides of A4 total, explaining the multiband compressor that you have created, and how the existing code was modified. Also describe any parameters used to control the effect. Include one or two plots showing the effect's operation. *Y*ou can use Audacity to apply your effect to a recorded file, then analyse the result in MATLAB to generate plots.

Helpful MATLAB Functions: *(use the 'help' function to learn more)*

`[y, fs, nbits] = wavread('filename')` *Read a WAV file from disk*

`wavwrite(y, fs, nbits, 'filename')` *Write a WAV file to disk*

`sound(y, fs)` *Play a vector as sound with sample rate fs*

`plot(), semilogx(), semilogy(), loglog()` *Linear and log-scale plot functions*

`fft(x, n)` *Calculate the FFT of length n. You might want to window the signal first. Also note that the fft is complex. Use* `abs(fft(...))` *to get the magnitude. A discussion of using fft() to generate spectra can be found here:*

http://www.mathworks.co.uk/support/tech-notes/1700/1702.html

`spectrogram(x, window, 'yaxis')` *Plot the spectrogram of x. If window is an integer, x is divided into that many evenly-spaced segments. 'yaxis' places frequency on the y axis and time on the x axis, as it often appears in audio analysis*

`(0:(length(x)-1))/fs` *For a signal x and sample rate fs, go from sample number to time. Useful for the x-axis of time-domain plots.*