Lab Report 2
Multiband Tremolo Effect

James Colla 440424255

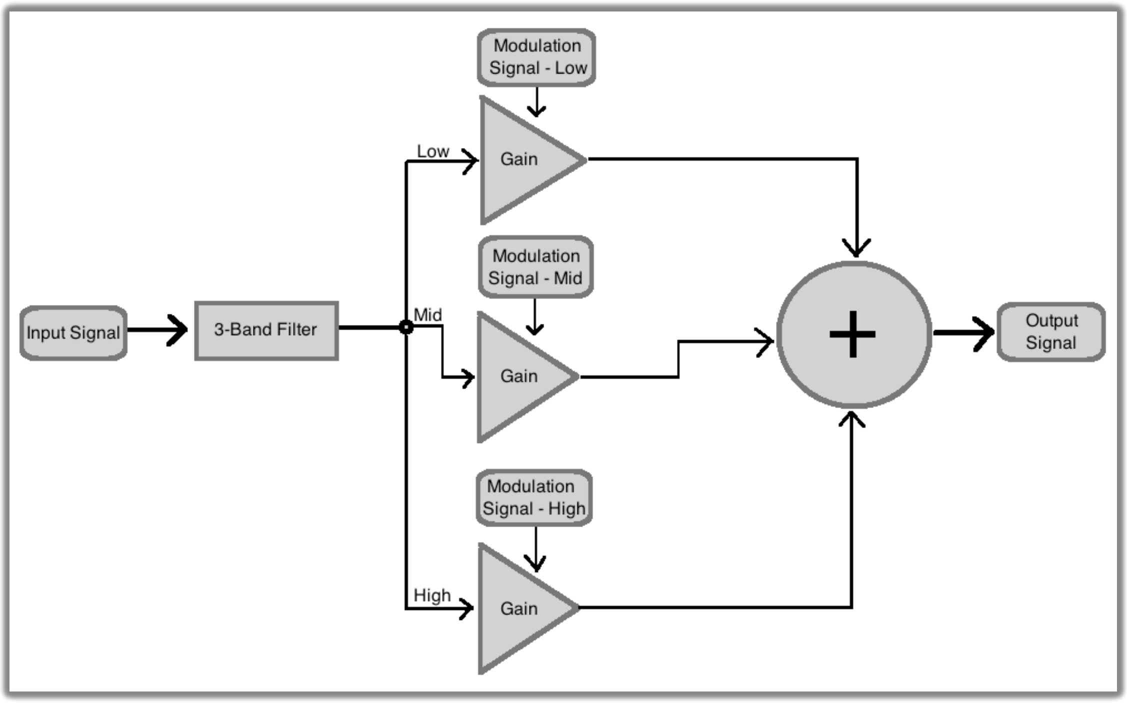
UNIT, DESC9115, Semester 1 2014
Masters of Design Science (Audio and Acoustics)
Faculty of Architecture, Design and Planning, The University of Sydney

Introduction

The Multiband Tremolo Effect created for this project would be intended for musicians or engineers to use on the music they create. It allows the musician or engineer to apply an individual tremolo effect to three bands within a sound recording. The Effect is designed to allow the person controlling the effect to change the rate at which the tremolo operates, the depth of each tremolo, the crossover frequencies between bands, and the mix of the affected signals against the original signal. This gives the operator full creative control of the effect. This type of effect does exist already in products such as Melda Production MMultiband Tremolo, as a VST effect plugin. This report looks to understand how such an effect would be created and implemented using Matlab.

The supplied script `CallScript` calls the function `MultiBandTremolo`, which in-turn calls the function `Tremolo`. Also supplied are one example of a clean electric guitar sound, and four versions of the sound after being affected by the `MultiBandTremolo` function.

The effect can be visualised in *Figure 1*, which is a signal flow diagram of the input signal through a filter and then the respective tremolos for each band.



The Calling Script

The supplied script `CallScript` first clears the workspace, as testing showed that not clearing the workspace caused problems with the construction of the filters. After this the supplied audio file `Guitar.wav` is imported to be the file run through the function. The function is then run with the imported audio file. After this, an array called `MBT_Output` is created, and written to a `.wav` file in the session folder. The script is then paused for one second, after which it plays the input audio file. Then there is a pause the length of the input file `x`, to pause the playing of the output file until after the input file is played.

MultiBandTremolo Function

The multiband tremolo function takes one input, stereo or mono, and runs it through a multiband tremolo effect. The output is an array with the same number of channels as the input. The function runs in four distinct sections:

1. Creation of the filters,
2. Filtered signals run through their corresponding tremolo functions,
3. Convolution of the resulting signals into the broadband effect signal
4. Mixing of the original signal with the affected signal.

Filter Creation

The filters for this project were created using the `fdesign` function in the three specifications: `fdesign.lowpass`, `fdesign.bandpass`, `fdesign.highpass`. These functions require the pass-band frequencies and stop-band frequencies to be defined. Since the function allows the operator to change the crossover frequencies, the pass-band and stop-band frequencies also need to change. To do this, equations were designed to have an appropriate response for the filter when the crossover frequencies are changed. The Band-pass Filter also required a numerator and denominator order defined for its IIR calculation. The following equations were used to determine the stop-band and pass-band frequencies of each filter:

Lowpass Filter

```
FpLow  = round(C1-(C1./10));
FstLow = round(C1+(C1./0.5));
```

Bandpass Filter

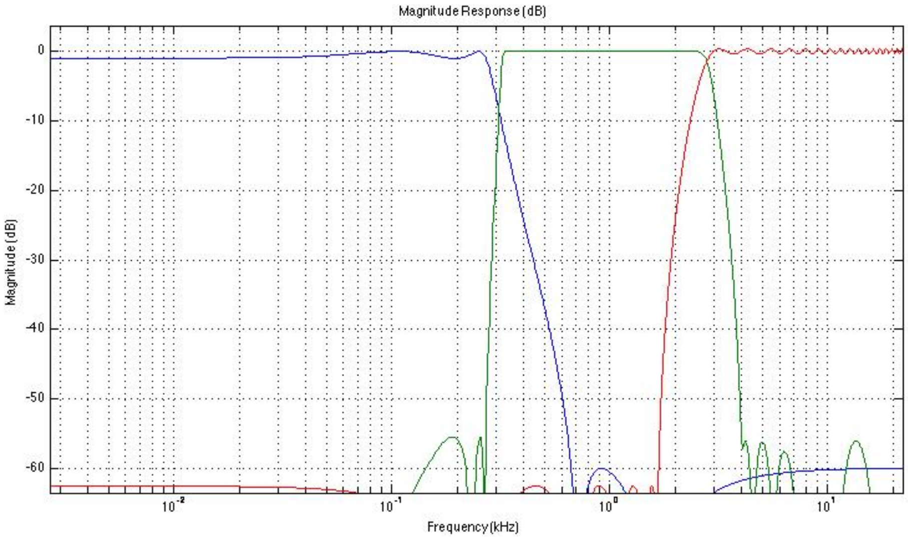
```
MidNb  = 50;
MidNa  = 10;
MidFs1 = (C1-(C1./10));
MidFp1 = (C1+(C1./10));
MidFp2 = (C2-(C2./10));
MidFs2 = (C2+(C2./1.5));
```

Highpass Filter:

```
FstHi  = (C2-(C2./3));
FpHi =   (C2+(C2./6));
```

The filters were then created using the `design()` function. The lowpass and bandpass filters used the IIR method, while the highpass filter was created using the FIR method.

The resulting filters led to a filter design that had a response very close to being flat. Due to the crossover frequencies of the filter being able to change, it was difficult to design filters that would provide a perfectly flat response. Figure 2 shows the response of the filter with crossover frequencies at 300Hz and 2500Hz.



Tremolo Design

Once the input signal is filtered into the three bands, each is used as the input for the Tremolo function, using the user-assigned parameters for the depth and rate of the tremolo.

The Tremolo function creates a modulation signal to apply to the input signal. To be able to convolve the modulation signal with the input signal, both signals need to be the same length. The length of the input signal is found by:

```
x_Length = length(x);
```

A for statement turns the vector into a cosine wave which uses the user-defined variables for depth and rate:

```
for x = 1:x_Length;
    TremSig(x)=(1-D)+D.*((1+cos((2.*pi.*R.*x)./fs))./ 2);
end
```

After this, the input signal is convolved with the output of the for statement TremSig, creating TremOut.

```
TremOut = x .* TremSig;
```

The last line of the function creates a column vector outsig, from the row vector TremOut. The outsig vector could remain a row vector, however since the input signal is a column vector, it is changed to keep the output as similar as possible to the input signal.

```
outsig(:,1) = TremOut(1,:);
```

Once each filtered signal is passed through Tremolo, they are normalised, just in case that during their run through Tremolo they amplified to have any points above or below 1.

```
LowTrem = LowTrem./max(abs(LowTrem));
MidTrem = MidTrem./max(abs(MidTrem));
HighTrem = HighTrem./max(abs(HighTrem));
```

Convolution and Output

The three signals are then convolved together to make the broadband signal TremSig, and then each channel is normalised.

```
TremSig = LowTrem.*MidTrem.*HighTrem;
TremSig(:,1) = TremSig(:,1)./max(abs(TremSig(:,1)));
TremSig(:,2) = TremSig(:,2)./max(abs(TremSig(:,2)));
```

After this, TremSig is scaled and mixed with the input signal at the user-defined level. After this it is normalised as a precaution.

```
outsig = (TremSig .* M).* (x .* (1-M));
outsig(:,1) = outsig(:,1)./max(abs(outsig(:,1)));
outsig(:,2) = outsig(:,2)./max(abs(outsig(:,2)));
```

The output of `MultiBandTremolo` (`outsig`) is then written to a new `.wav` file in the session folder.

```
wavwrite(outsig,fs,'MBT_Output');
```

Difference Between Mono and Stereo Inputs

`MultiBandTremolo` is designed to work with both mono and stereo input signals. To have this functionality, `MultiBandTremolo` is required to perform one way for a mono input, and a second way for a stereo input. What has been mentioned about `MultiBandTremolo` previously is how the function runs with a mono input.

The only difference between the two treatments is that stereo input signals are split into two mono signals (left and right channels, `SigL` and `SigR`). After this, each channel is treated the same as if it is its own mono input signal. After passing through `Tremolo`, the left and right channels are brought back together to form a stereo signal.

The function determines whether the input is mono or stereo using an `if` statement, where the `size()` function helps determine the number of input channels.

Evaluation and Future Developments

The effect created does work successfully, and as intended from the beginning. The function is easy to use and understand, and works in an efficient manner for what is being done.

`MultiBandTremolo` does not have the same amount of customisation as the Melda Productions product, the most important being the ability to change the shape of the tremolo modulation signal. By having a pure cosine wave modulating across three bands at three different rates, the waves are able to easily combine and have a dramatic amount of adding and cancelling. By having imperfect cosine or sine waves, the Melda Productions product allows the three tremolos to work more independently of each other, and not always combine or cancel each other perfectly. Other than this, `MultiBandTremolo` has the same amount of creative control as the MMultiband Tremolo VST plugin, given it is being implemented in Matlab.

Example Outputs

File Name	R1	D1	C1	R2	D2	C2	R3	D3	Mix
MBT_Output	5	0.7	300	7	0.6	2500	9	0.7	0.8
MBT_Output1	2	1	400	6	1	2100	9	1	0.5
MBT_Output2	12	1	250	12	1	2000	12	1	0.3
MBT_Output3	8	0.6	280	3	0.9	1960	8	0.75	0.2
MBT_Output4	6	0.2	500	6	0.2	3600	10	0.8	0.4

References

Apply design method to filter specification object - MATLAB design - MathWorks Australia. 2014. Apply design method to filter specification object - MATLAB design - MathWorks Australia. [ONLINE] Available at: http://www.mathworks.com.au/help/signal/ref/design.html. [Accessed 25 May 2014].

Bandpass filter specification object - MATLAB fdesign.bandpass - MathWorks Australia. 2014. Bandpass filter specification object - MATLAB fdesign.bandpass - MathWorks Australia. [ONLINE] Available at: http://www.mathworks.com.au/help/signal/ref/fdesign.bandpass.html. [Accessed 17 May 2014].

Filter specification object - MATLAB fdesign - MathWorks Australia. 2014. Filter specification object - MATLAB fdesign - MathWorks Australia. [ONLINE] Available at: http://www.mathworks.com.au/help/signal/ref/fdesign.html. [Accessed 17 May 2014].

Highpass filter specification object - MATLAB fdesign.highpass - MathWorks Australia. 2014. Highpass filter specification object - MATLAB fdesign.highpass - MathWorks Australia. [ONLINE] Available at: http://www.mathworks.com.au/help/signal/ref/fdesign.highpass.html. [Accessed 17 May 2014].

Lowpass filter specification - MATLAB fdesign.lowpass - MathWorks Australia. 2014. Lowpass filter specification - MATLAB fdesign.lowpass - MathWorks Australia. [ONLINE] Available at: http://www.mathworks.com.au/help/dsp/ref/fdesign.lowpass.html. [Accessed 17 May 2014].

Methods available for designing filter from specification object - MATLAB designmethods - MathWorks Australia. 2014. Methods available for designing filter from specification object - MATLAB designmethods - MathWorks Australia. [ONLINE] Available at: http://www.mathworks.com.au/help/signal/ref/designmethods.html. [Accessed 25 May 2014].

MMultiBandTremolo. 2014. MMultiBandTremolo. [ONLINE] Available at: http://www.meldaproduction.com/plugins/product.php?id=MMultiBandTremolo [Accessed 12 May 2014].