MPATE-GE2599
Fundamentals of Digital Signal Theory
Dr. Agnieszka Roginska
Michael D. Quigley
12/18/11

I. Overview

The goal of my final project was to construct a *Ring Modulation and Short Time Fourier Transform* function in MATLAB. Motivations behind the project came from the extensive use of ring modulation in musical production and sound design for film and television, particularly in the voices of the Daleks in the long-running BBC series *Doctor Who*. Interest in the Fourier transform came from our discussions in class and outside research about the sinusoidal construction of audio signals and the algorithmic transformation from time domain to frequency domain representations, and vice versa, without information loss, as discovered by Jean-Baptiste Joseph Fourier. The resulting MATLAB function, titled *ringmod*, takes in a number of inputs specifying modulation signal, carrier frequency and amplitude, window and overlap sizes, window type, and output filename. It then modulates the signal with a sine wave specified by the carrier frequency, writes the output to a new .WAV file, performs a short time Fourier transform on the output, plays back the ring-modulated output, and graphs the resulting frequency and amplitude data. The two main challenges for the project were the conceptualization of the ring modulation portion of the code, and the adaptation of the short time Fourier transform portion of code from a previous lab assignment into the context of the *ringmod* code. Although the concept of the short time Fourier transform code was preserved for this project, it was necessary to alter several aspects of it to properly fit the goals of the *ringmod* function.

Ring modulation derives its name from its corresponding analog electronic circuit, in which a series of four diodes form a bridge circuit, much like a bridge rectifier but with the diodes all facing in a clockwise or counter-clockwise direction, anode to cathode (Orton). As a form of amplitude modulation, it is implemented digitally by multiplying two signals in the time domain, resulting in the sum and difference frequencies of the carrier and modulator signals. This mixing of the frequency components of each input to yield higher and lower frequency components is known as heterodyning (Moore, 47). Amplitude modulation was used extensively in radio broadcast technology as a way to transmit over long distances. Several methods of transmission have developed over the course of radio communication history, but amplitude modulation is perhaps the simplest. As is stated by the law of convolution, multiplication in the frequency domain is equivalent to convolution in the time domain. This relationship is interesting because the reverse is also true, that multiplication in the time domain is equivalent to convolution in the frequency domain. This is of particular importance to this project because the time-domain product of the carrier and modulator signals can be represented as the frequency-domain convolution of the two signals. In the time domain, the constant amplitude of the carrier signal is shaped by the instantaneous amplitude of the modulator to form an output waveform whose envelope resembles the modulator but is comprised of signals that are the sum and difference frequencies of the carrier and modulator. By doing this, we effectively shift the modulator signal to a range above and below the carrier frequency. This is important for radio communication because it allows for the shifting of an audible audio signal into a range beyond human perception (radio bandwidth) for transmission that is then shifted back down to the audible range at the receiving end. By looking at the process from the frequency domain, we have essentially convolved the frequency spectrum of the modulator signal with the frequency spectrum of the carrier signal (Smith, 204-205). Much like convolution in the time domain, convolution in the frequency domain can yield sonically interesting results.

Ring modulation as a form of amplitude modulation tends to utilize a sinusoid as a carrier signal and a more complex waveform for a modulator signal. Important to note also is the range in which ring modulation occurs. As a popular audio effect, the frequency of the carrier signal must be within the range of human perceptibility. Too high a carrier and the modulated output will go above our range of hearing. Consideration of the sampling rate is also an issue to consider when dealing with digitally implemented ring modulation. As the carrier frequency approaches the Nyquist limit, component frequencies of the output that are above the Nyquist will be aliased and present themselves as lower

frequencies in the signal.This can either be an unwanted artifact of a system or an interesting sonic aspect to digital ring modulation. As an effect, it is often used to create sounds categorized as metallic. This has to do with the non-harmonic partials that result from the modulation. As a modulator signal increases in complexity, by definition the number of partials in the signal increase. This means that each partial is modulated with the carrier frequency, thus producing new frequency components that are not related to the harmonic series of the original signal. Very quickly the output of the ring modulator increases in non-harmonic complexity. This is what is associated with metallic sounds. A metallic timbre is rich in non-harmonic partials, and so as a voice is fed through a ring modulator, for example, the output is a metallic, robotic-sounding voice, much like that made famous by the Daleks in *Doctor Who*.

The British Broadcast Channel credits the sound design of the Daleks' voices to Brian Hodgson, a composer and sound technician that worked for the BBC Radiophonic Workshop in the 1960s. According to his artist biography, Hodgson created the voices “by distorting the actors' voices and feeding them through a device called a ring modulator” (BBC). This period in the 20th Century saw the application of audio processes in popular mediums that had previously existed only in avant-garde electronic music studios. Only a few years before Hodgson's work for the BBC, Louis and Bebe Barron had composed the first electronic score for the feature film “Forbidden Planet” in 1956 (Hevesi). The score featured extensive use of analog electronic circuits constructed by the duo that included several ring modulators. Later, works by Karlheinz Stockhausen like *Mixtur* (1964) and Telemusik (1966) featured extensive use of ring modulation to process the audio content of both acoustic instruments and tape recordings (Orton). I felt that the history of ring modulation was particularly interesting, and it's extensive and continued present day use in music and sound design drove me to design my project around the process. I have long been a fan of early science fiction and the electronic music boom of the 20th Century fascinates me. Naturally, constructing a project around such a time-tested, iconic audio effect seemed like a great idea.

The second part of the *ringmod* function is the short time Fourier transform. My interest in this topic also stems from 19th and 20th Century innovations in sound. Jean-Baptiste Joseph Fourier was a French mathematician and physicist active in the late 1700s and early 1800s. His work with heat propagation lead to paper about sinusoidal representations of temperature distributions. The importance of this paper is the claim that any periodic signal can be represented by the sum of specific sinusoidal waves (Steven Smith, 141). This claim was ground breaking and would go on to completely change the development of signal analysis and synthesis. From Fourier's hypothesis would stem a number of methods of sinusoidal decomposition. What would become known as the Fourier transform was the concept of decomposing a continuous periodic signal into its component frequencies. A number of algorithmic functions were created that would allow anyone to analyze a given continuous periodic signal and be given the frequency, phase, and amplitude values of each component sinusoid. Over time methods were developed for the analysis of continuous or discrete and periodic or aperiodic signals (Steven Smith, 144). With these developments came the birth of electronic music synthesis. The ability to analyze an audio signal to discover its core components meant that the signal was reconstructable by synthesizing it's exact components. This is one of the most important concepts in the development of electronic music in the 20th and 21th century. With the increasing development of computational power, it has become faster and less resource-heavy to perform analysis on complex audio signals and effectively re-synthesize them. Personally, I feel that Fourier Analysis is one of the most interesting developments, specifically regarding sound and music, in scientific history. Because of this I felt it would be of interest to those taking the time to modulate a signal to see the frequency components of their ring-modulated output. To see the components of a complex audio signal is like examining a cell with a microscope. These are the building blocks from which the sounds that fascinate us are made of. The Fourier transform in the function can serve as an education tool for displaying the building blocks that make up the output of the ring modulator.

II. Signal Processing

For the *ringmod* function, I needed to design a system of digital signal processing that would modulate a desired audio signal with a sinusoid and output the result. I decided that the user should be able to determine the frequency and amplitude of the sinusoid so that a nearly limitless amount of modulated audio outputs would be possible. As such, the audio signal and the sinusoid, or carrier signal, are multiplied in the time domain. This results in a number of component frequencies at the sum and difference of the modulator and carrier frequencies. For example, if the modulator signal was a sinusoid at 40Hz and the carrier signal a sinusoid at 60Hz, the resulting signal would be comprised of a 20Hz sinusoid and a 100Hz sinusoid. As the modulator signal increases in complexity, a new component frequencies is created at the sum and difference of the carrier plus and minus each partial. For example, a modulator signal with a fundamental frequency of 60Hz and a partial at the 1st harmonic, 120Hz, modulated with a carrier signal at 200Hz would result in an output signal whose component frequencies are 140Hz, 260Hz, 80Hz, and 320Hz. It is easy to see how quickly the results increase in complexity. The amplitudes of each component frequency in the output is dependent on the amplitudes of the modulator and carrier signals. At this point in the process, the output data is sent to a function that writes a new audio file; it is also sent to a short time Fourier transform. The short time Fourier transform, in this case a discrete STFT, is the "succession of [fast Fourier transforms] of windowed data frames, where the window 'slides' or 'hops' forward through time." (Julius O. Smith). In our case, the ring modulator output is windowed, FFTs are performed on each successive window, and resulting data matrix is analyzed to extract the amplitude data for the component sinusoids. A frequency vector is generated and plotted against this amplitude data to give the user a graph that clearly shows the component frequencies and their relative amplitudes. The output of the ring modulator is then played back in the time domain, at the same sampling rate as the modulator signal, for the user to hear his or her result. What follows is a more in depth description of how exactly this entire process is implemented using MATLAB.

III. Implementation

The function takes in the following inputs: *signal* - the modulator audio file, *fCarrier* – the frequency of the carrier signal, *aCarrier* – the amplitude of the carrier signal, *winLength* – the window length of the short time Fourier transform, *overLapLength* – the length of the overlap for each window, *winType* – the type of window used in the short time Fourier transform, and *OUTfile* – the filename to be written for the output ring-modulated audio file. Provided the signal is a .WAV file, the function reads in the audio data, sampling rate, and number of bits. It is then analyzed to find it's length in samples and the number of channels of audio. If it is found to have more than one channel, the function monos the signal by taking its mean. At this point in the function, a time vector is generated from the signal length and sampling frequency. A carrier sine wave is constructed from the time vector and the specified frequency and amplitude, that is then multiplied by the modulator signal. This multiplication is the digital equivalent of ring modulation in analog electronics. The product of these two signals, the ring-modulated signal, is normalized to ensure there is no clipping upon playback, and is then written to the output audio file using the MATLAB function "wavwrite". Wavwrite is given the normalized output signal, the initial sampling rate of the input, the initial number of bits of the input, and the filename specified by *OUTfile*. At this point the ring modulation aspect of the code is finished, and it enters the short time Fourier transform section.

The first portion of the short time Fourier transform is the calculation of the hop size, the variable that determines at which sample a new window occurs. It is found by subtracting the overlap length from the window length. The output from the ring modulation section is then zero-padded with one additional window length of zeros. The window type is then determined using a switch case block.

The window type is analyzed to determine which of the five acceptable window types is supplied. Depending on which window type the user specifies, the function sets the *window* variable to the appropriate type of window and length. If the user does not supply an appropriate window type, the function returns an error with a list of the appropriate window types that they may enter. This function only allows window types of the following: Rectangular, Hamming, Hann, Blackman, and Bartlett. Once the window type is determined and a window of length and type specified is constructed, a variable representing half the length of the window is initialized. An empty matrix, *y*, is then initialized for the data that is to be supplied by the short time Fourier transform, with the number of rows as half the window length and the number of columns as the quotient of the signal length and the hop size. A variable *column* is then initialized at 1. All this is in preparation for the for-loop that generates the short time Fourier transform data. An index is set from 1 to the length of the signal in increments of the hop size. The *window* variable is then applied to the zero-padded ring-modulated signal for the current column. The fast Fourier transform of the windowed signal is placed into the output matrix *y* for the current column, and the column variable is increased by one. The for-loop then repeats for the new column, until the end of the signal is reached. The matrix *y* is now filled with the complex data output of the fast Fourier transform function for each column. The absolute value of the output is taken to extract the magnitude values. These values are normalized and then set to the y-axis of the graph. A frequency vector is constructed for each magnitude value and is set to the x-axis of the graph. The function then sounds the output audio from the ring modulation code and finally graphs the frequency vs. magnitude data on a graph titled “Frequency vs. Magnitude of Output”.

IV. Conclusions and Future Work

I consider my project to be successful; as a function it meets the goals I had intended for it. It can produce sonically interesting results depending on the user's inputs, it provides a visual, and perhaps educational representation of the output to better the user's understanding of the result, and it does so in a relatively efficient manner. There is no limit to the length of the audio input other than the computational power of the machine running MATLAB. Although it is important to note that a complex waveform that changes over time will quickly fill the graph with aesthetically pleasing but not very useful frequency data. I designed this project because of my interests in the use of ring modulation in sound design and the power of Fourier transforms. As a result of actually building the function, I ended up researching the two concepts in depth and became even more intrigued. It was interesting for me to learn about the history of Jean-Baptiste Joseph Fourier and his mathematical contributions to the world as they apply to music technology, and to learn about the history of ring modulation and its place in the electronic music boom of the 20th Century. I have always been fascinated by the events that shaped the current state of music technology, and so to spend some time with an audio processing method that was new and exciting for people like the Barrons and Karlheinz Stockhausen was a unique learning experience. The computational power that we have at our fingertips today makes experiencing audio processes like ring modulation just a few clicks away. This makes me appreciative, but also nostalgic for era before my time when such processes were new, exciting, and even controversial. The Barrons were denied credit as composers because the electronic score they created for “Forbidden Planet” would not be considered music by the American Federation of Musicians (Hevesi). Today I can create such sounds on my portable laptop computer with little to no cost. The advances in digital signal processing are astounding, and working on the *ringmod* function allowed me to experience just a small aspect of the power of DSP. In the future I think constructing an analog ring modulator may be of interest to me. Observing the parallels between analog- and digital-domain electronics can often result in a better understanding of the underlying concept. And realizing the limitations of each domain is a useful insight. I personally enjoy working in the analog electronic domain, and would be interested in building a ring modulator for real-time use. I know that real-time ring modulators are achievable in

digital signal processing as well, but actually building a piece of hardware with a dedicated purpose might be a useful addition to my voice and guitar performance rig. Hopefully I can look to the experience of processing digital audio signals in MATLAB to help in future projects as well. It would be helpful to prototype various effects systems in MATLAB to see what is sonically interesting before choosing to invest in building a physical unit.

Bibliography

BBC. "Brian Hodgson." BBC Music -Artist Biography. *Wikipedia.* Web. 18 Dec. 2011. <http://www.bbc.co.uk/music/artists/bacf0e5c-6550-4f64-b8f3-def40d7a66ed>.

Hevesi, Dennis. "Bebe Barron, 82, Pioneer of Electronic Scores, Is Dead - New York Times." *The New York Times - Breaking News, World News & Multimedia*. The New York Times Company, 25 Apr. 2008. Web. 18 Dec. 2011. <http://www.nytimes.com/2008/04/25/movies/25barron.html?_r=2>.

Moore, F. Richard. *Elements of Computer Music*. Englewood Cliffs, NJ: Prentice Hall, 1990. Print.

Orton, Richard. "Ring modulator." Grove Music Online. *Oxford Music Online*. 18 Dec. 2011 <http://www.oxfordmusiconline.com/subscriber/article/grove/music/23491>.

Smith, Julius O. "Practical Computation of the STFT." *Center for Computer Research in Music and Acoustics | CCRMA, Stanford University*. W3K Publishing, 2011. Web. 18 Dec. 2011. <https://ccrma.stanford.edu/~jos/sasp/Practical_Computation_STFT.html>.

Smith, Steven W. *The Scientist and Engineer's Guide to Digital Signal Processing*. San Diego, CA: California Technical Pub., 1997. Print.