Translating a Digital Signal Modulation Synthesizer from LabVIEW to Simulink

Drew M. Canfield, Kristi M. Perreault, Joseph D. Legris, Benjamin D. McPheron
School of Engineering, Computing & Construction Management
Roger Williams University

Sound synthesizers have a broad range of uses for signal modulation and musical manipulation. Music utilizing synthesizers has been popular for many decades, and has seen resurgence in recent years in both the music and film industries. Some engineering educators have seen the value of implementing sound synthesizers to help students explore signals and systems concepts. Currently, there is literature describing a synthesizer implemented in LabVIEW by Ed Doering, but the modification and distribution of his software is limited to the LabVIEW programming environment. While LabVIEW is a useful, accessible, and powerful tool, it lacks some capabilities afforded by MATLAB and Simulink for post processing data. Currently there is not documentation of a similar synthesizer implementation existing in the MATLAB and Simulink environment. Thus, this project applies the Simulink package in MATLAB to create a digital synthesizer capable of audio signal manipulation. By using the signal processing toolbox available through MATLAB, a synthesizer was created by translating LabVIEW models to Simulink equivalents. The resulting synthesizer is capable of frequency and amplitude modulation synthesis, additive synthesis, and subtractive synthesis. In addition, a basic graphical user interface was developed that allows for modular combination of each digital signal processing component. This paper details the development and operation of the Simulink synthesizer so that educators and researchers can replicate this approach.

Corresponding Author: Benjamin D. McPheron, bmcpheron@rwu.edu

1. Introduction

Audio synthesis has existed for a number of years, with roots dating back to as early as 1860s in the form of analog synthesizers. Over the years, these synthesizers have grown exponentially from their early beginnings as just modest single sine wave oscillators and white noise generates. Synthesizers expanded to include many new technologies, and innovation continued even further to the more recent device, the digital synthesizer [1].

In today's world, synthesis refers to any number of sound filters and modules employed in a variety of environments, including the common engineering software tool, LabVIEW. Using this programming environment, Stokes and Doering [2] generated a digital music synthesizer, which included filters such as vibrato and frequency modulation, in addition to a complex user interface. This implementation has served as an excellent teaching tool for instructing signal processing.

Unfortunately, there are no documented audio signal synthesizers developed in the MATLAB and Simulink environment. One benefit of MATLAB is the advanced data visualization capabilities. As a result, this paper mimics the implementation by Stokes and Doering [2], but instead uses the MATLAB and Simulink environment. The modules that have been implemented include frequency and amplitude modulation, additive synthesis, subtractive synthesis, and chorus synthesis, and a well-defined graphical user interface.

2. Background

2.1 Modulation Synthesis

Frequency modulation (FM) remains one of the most common types of sound synthesis. Prior to Chowning in 1977 [3] the primary use of FM was for radio transmission. Chowning describes FM synthesis as one of the most simplistic methods to creating complex spectra, as it can be easily manipulated and controlled. In his research, Chowning defines the most prominent parameters of FM synthesis to be a carrier frequency, a modulating frequency, and a peak deviation. When these aspects are combined, the modulating frequency alters (modulates) the carrier frequency to output the filtered sound, whose parameters are defined by the input's frequency and modulation [3].

Amplitude Modulation (AM) is similar to FM synthesis, but differs in the way that waves are modulated. Similar to FM, AM consists of a carrier wave and a modulating wave, however, the modulating wave modulates the amplitude of the carrier wave, rather than the frequency

[4]. The output wave is defined by the input's amplitude and modulation.
Both frequency and amplitude modulation are implemented in the digital synthesizer created in MATLAB and can be applied to any audio signal.

2.2 Additive Synthesis

Additive synthesis refers to a number of different synthesis techniques, all of which rely on the idea that complex tones can be created through the summation of simpler tones [5]. One historic example of analog additive synthesis is an organ, which combined the sounds generated from separate pipes to create a pleasant, complex noise. One modern example could be the formation of a sawtooth function, which can be approximated through the summation of harmonics to an original, fundamental signal with amplitudes equal to the inverse of the partial being added. The individual components and the resulting signal can be seen in Figure 1 and Figure 2.

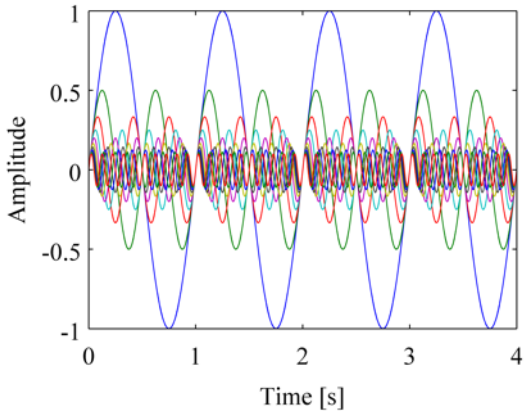

Figure 1: Each individual harmonic plotted against each other for visualization

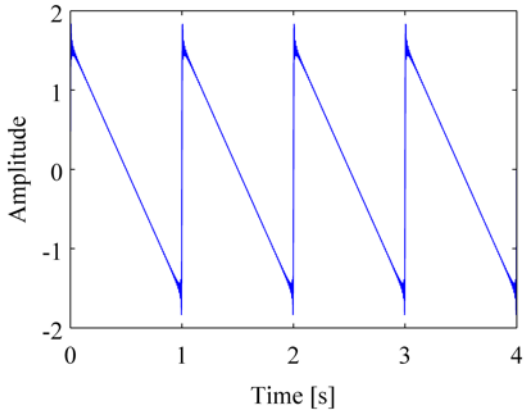

Figure 2: The result of the summation of each harmonic to form a square wave

Previously, additive synthesis was a difficult processing step due to the large amount of data generated per harmonic added [6,7], however, with increases in computing power, it has become possible for additive synthesis to be plausible. Doering [2] developed one such digital synthesizer capable of multiple filter types including additive synthesis using the software package LabVIEW. Using similar techniques, an additive synthesizer was developed in MATLAB.

2.3 Subtractive Synthesis

An adequate synthesizer should include an aspect that performs 'subtractive' synthesis on any given signal. A subtractive synthesizer is most commonly categorized as the alteration of timbre within a sound after succumbing to low-pass filters [8]. The characteristic of timbre is what makes two sounds different, even if they share the same loudness and pitch. When a signal passes through a low-pass filter, it will return a signal with a lower frequency because it attenuates higher frequencies. The threshold in which signals are either passed or attenuated is known as the 'cutoff frequency', and this is the value that controls the characteristics of the filter [9].

For this specific application of low-pass filters, a filter known as a 'Butterworth filter' is implemented. Butterworth first introduced this type of filter in the 1930s in his paper titled "On the Theory of Filter Amplifiers" [10]. The Butterworth filter can act as either a high-pass filter or low-pass filter, depending on certain parameters. For this application, a low-pass filter is created to implement subtractive synthesis. The Butterworth filter has a passband that is incredibly flat until the cutoff frequency, then the gain rolls off toward a factor of zero. This filter maximizes flatness in comparison to other types of low-pass filters, and minimizes ripples within the response [10]. Figure 3 shows how the magnitude of a given signal will decay when a discrete low-pass filter is implemented. The cutoff frequency is 100 Hz, and it can be seen that the magnitude of the signal drops off at that frequency.

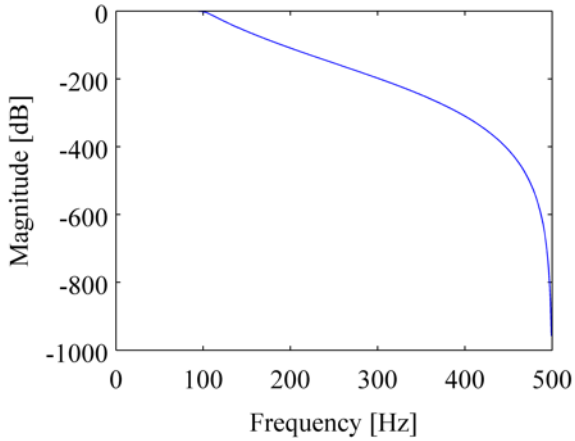

Figure 3: Subtractive signal decay

The synthesizer in this work incorporates the subtractive synthesis by means of discrete filters within Simulink holding Butterworth filter parameters values. This form of synthesis can be chosen within a graphical user interface and implemented by passing any generated signal through the filter.

3. Methodology

3.1 Modulation Synthesis

As previously defined, FM synthesis consists of two sine waves know as the modulator and the carrier, which are generated and combined, producing a filtered wave whose properties are determined by their frequencies and modulation degrees. The frequency of the carrier wave is modulated by the modulation wave with a frequency ω_m, as shown by

$$X(t) = A(t)\sin\{\omega_c t + I(t)\sin(\omega_m t)\}$$

Eq. 1

To create this filter in Simulink, the user-defined frequency and amplitude is uploaded from the MATLAB workspace to the Simulink model, where the signal traveled through a discrete time integrator to transform the system output to purely discrete values [11]. A gain block was then used to multiply the signal by a constant to amplify. A simple ramp function was added to the signal, and this output was sent to a sine function defined as the carrier wave, and the final filtered signal was outputted to the user interface. Although the user can only define the values of amplitude, frequency, and duration, the programmer may alter the constant values and Simulink parameters to their preference.

AM synthesis had a slightly different implementation. Simulink proved to be an accurate platform for FM synthesis, however, when used for AM synthesis, the interface proved to cause a large time delay. In order to improve synthesizer efficiency, the AM filter was created solely through MATLAB. The sine wave generated by the user-inputted frequency and amplitude was read in and a constant value was added. A separate sine wave (the carrier wave) was generated, and the product of these two waves created the AM synthesized wave.

3.2 Additive Synthesis

Rather than develop a Simulink model, a MATLAB function was developed to generate multiple signals of input amplitude and frequency. Each of these signals were added to the original signal, either an internally generated sound with user input parameters or an uploaded audio signal. A user has the option of selecting the use of the additive synthesizer or not with up to 5 harmonics. The MATLAB code could easily be expanded in the future for any number of harmonics.

3.3 Subtractive Synthesis

The Butterworth filter in MATLAB creates a standard difference equation, which is associated with the z-domain transform of the system as shown in equation 2.

$$G(z) = \frac{B_1 + B_2 z^{-1} + \cdots + B_n z^{,-n}}{A_3 + A_4 z^{-1} + \cdots + A_n z^{-n}}.$$

Eq. 2

The z-domain transform is used within the Simulink file that is generated. The coefficients B_i and A_i are input into the discrete filter block in Simulink to create an equivalent Butterworth filter block. Once the user inputs how many filters they want to attenuate the signal through, the user must specify filter order. The higher the order the sharper the filter cutoff; however, if the order is too high, the system will become unstable and not return a signal. The user will also specify the cutoff frequency for each of the Butterworth filters.

The subtractive synthesizer implements a large range of combinations of low-pass filters, as well as an ability to alter each filter individually by changing the cutoff frequency. This synthesizer can alter any signal passed through by attenuating any frequency higher than the specified cutoff frequency, successfully creating a subtractive synthesis generation.

3.4 Graphical User Interface (GUI)

A GUI was developed in MATLAB for a user to navigate to use the developed digital synthesizer in this paper. The GUI was designed to allow a user to select whether to generate an original fundamental signal or upload a .wav file, and send it through any of the filters specified prior. The final GUI can be seen in Figure 4 (next page)**Error! Reference source not found.**.

If a user specifies that they would like to upload an audio file rather than generate a signal, a message box prompts the user to adjust the GUI code to add the file name to be uploaded prior to running the synthesizer. The final result of running the synthesizer is the output of the new signal through the speakers of the device running the simulation, the creation of two sound files (one of the original signal, one of the synthesized signal), and a plot showing the original and synthesized signals.

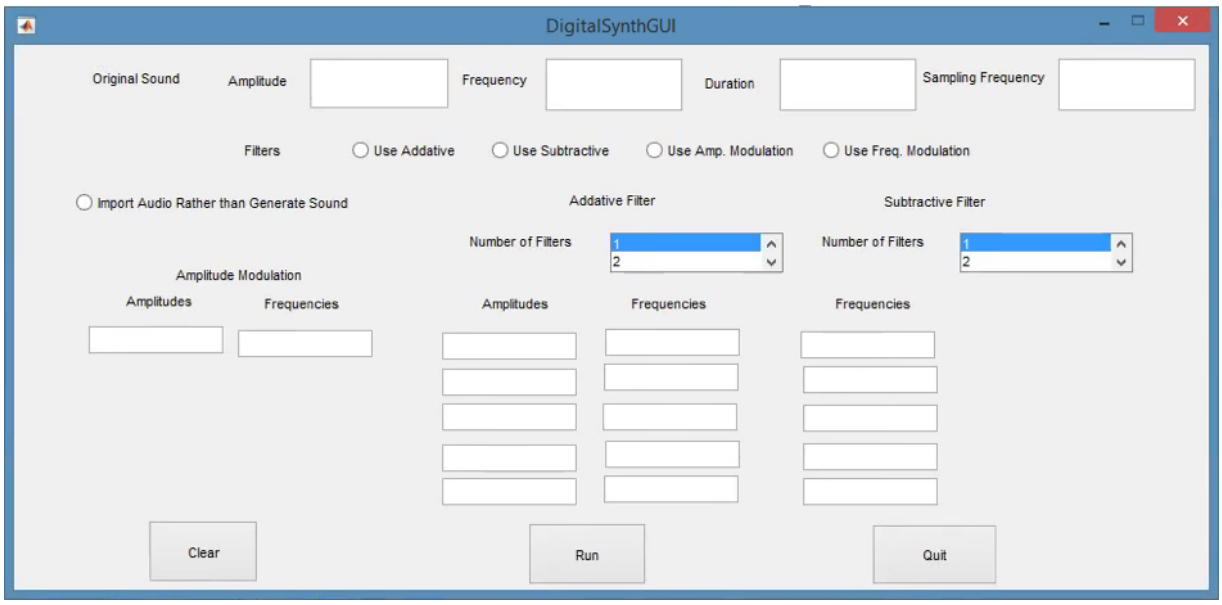

Figure 4: GUI for developed digital synthesizer

4. Analysis

4.1 Modulation Synthesis

The FM synthesis proved to be successful, as it accurately generated an FM synthesized signal from a user defined sine wave. Figure 5 depicts both the user-inputted wave and the altered output wave modeled at a sampling frequency of Fs. As shown in the figure, the frequency of the programmatically defined carrier wave is modulated to produce the output shown.

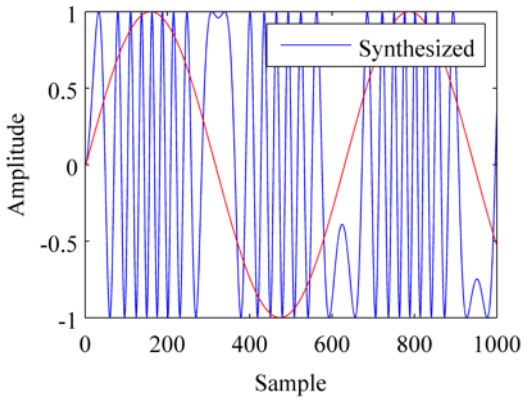

Figure 5: FM Modulated signal compared to original signal

The frequency of half of the period of the wave has increased as illustrated by the condensed and more frequent oscillations, and decreased over the second half of the period, as indicated by the fewer amount of oscillations.

In addition, the AM synthesis proved to be just as accurate. As shown in Figure 6, the original wave is modulated to produce a sine wave with a much varying amplitude.

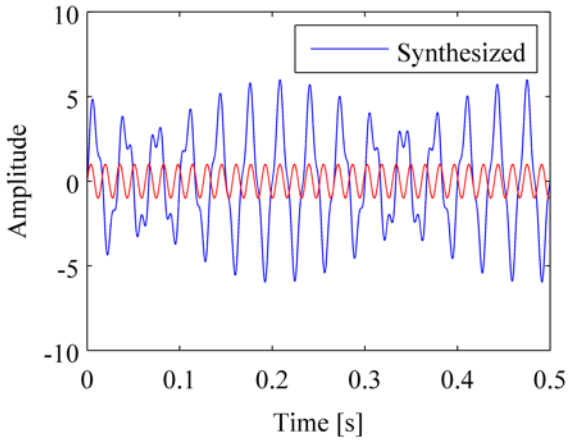

Figure 6: Amplitude modulated signal compared to original signal

The frequency and period remained roughly the same, making the only change that of the amplitude, which increased overall in terms of the original signal and continued to vary throughout. This change in the amplitude of the new wave can be modeled by a sine wave, as seen by the variation of the amplitude in the outputted wave. Although there is some discrepancy in the modified wave at its lower amplitudes, it is minor and can be considered negligible for these purposes.

4.2 Additive Synthesis

Implementation of two harmonics with the following parameters were added to a fundamental signal with

amplitude of 1, frequency of 400 Hz, duration of 4 seconds, and sampling frequency of 44100 Hz; 30 Hz frequency with amplitude of 1 and 110 Hz frequency with amplitude of 0.5. A plot of the original and synthesized signal can be seen in Figure 7.

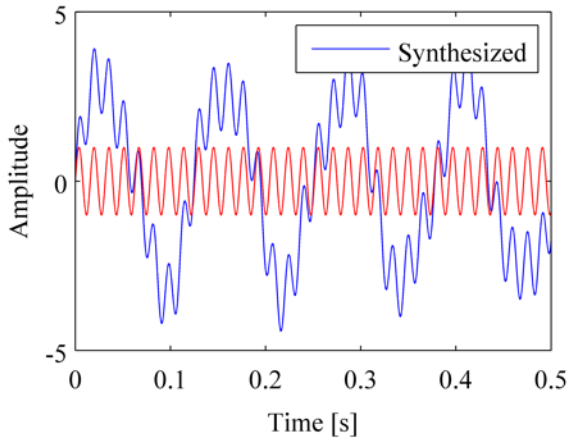

Figure 7: Plot of the synthesized original sinusoidal signal sent through additive synthesizer with two harmonics

Again, the program is capable of using up to five harmonics.

4.3 Subtractive Synthesis

The subtractive synthesis portion of the MATLAB Simulink synthesizer was successful as it can alter an input signal to a signal that has been attenuated based on a specified cutoff frequency. Within the graphical user interface, the user must input certain parameters for the signal that is generated, the number of filters that they want the signal to pass through, and the cutoff frequency of each filter that the signal passes through. Figure 8 below depicts a generic signal generated by the synthesizer against the same signal passed through a single Butterworth filter. The signal is operating at a sampling frequency of 1000 Hz and has succumbed to a cutoff frequency of 20 Hz.

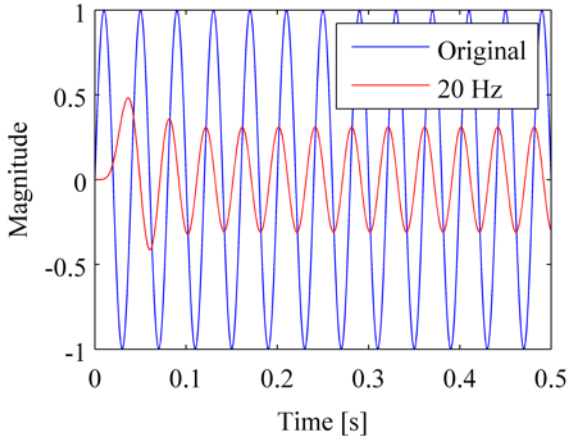

Figure 8: Signal sent through Butterworth filter compared to original signal

When the signal is passed through numerous filters, it is important to note that each consecutive filter must have a lower cutoff frequency than the previous in order to acquire results. If they increase instead, the first filter will attenuate the higher frequencies and the following filters will not change the signal at all. Figure 9 depicts the same generic signal from Figure 8 against a signal that has passed through two consecutive Butterworth filters. The signal is operating at sampling frequency of 1000 Hz and has succumbed to cutoff frequencies of 15 Hz.

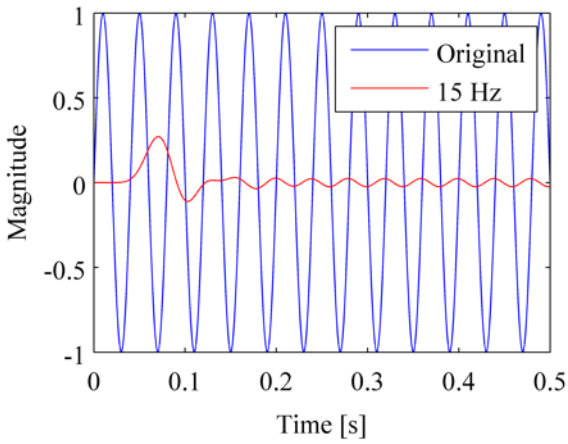

Figure 9: Signal passed through two Butterworth filters compared to original signal

It is seen within the figures that the signal becomes weaker, and the peaks become less pronounced. This is due to the low-pass filters only allowing certain frequencies below the cut off frequency to pass through. If the cutoff frequency was increased, the altered signal would look more like the original signal. If the cutoff frequency is decreased, the peaks would get even smaller. This is where the user has the most control of applying the subtractive filter. Figure *10* gives a better depiction of the cutoff frequencies, and shows a basic sawtooth wave being altered by a single Butterworth filter at differing cutoff frequencies. It is worth noting how the edges of the sawtooth become less rigid and the signal becomes less responsive when the filter is applied. For this example, the sampling frequency is taken at 1000 Hz, and the cutoff frequency decreases from 100 Hz to 25 Hz. At a cutoff frequency below 25 Hz, the signal becomes essentially attenuated.

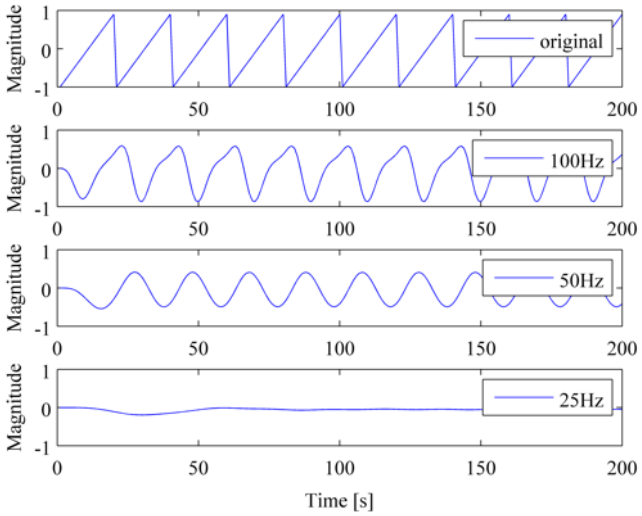

Figure 10: Cutoff frequencies

5. Conclusion

Using the work by Stokes and Doering [2] as a reference, a working digital synthesizer with a graphical user interface was created using MATLAB and Simulink. The code and corresponding Simulink files were created for amplitude modulation, frequency modulation, additive synthesis, and subtractive synthesis, and were deemed not only successful but also accurate in producing the expected output.

There currently exists literature on a number of different types of filters and syntheses that were not explored in this implementation due to time and resource constraints. If this project were to be replicated, it would be beneficial to research and implement additional filters, and to continue exploring the other commands and toolboxes that MATLAB has to offer.

6. References

1. Review of Sound Synthesis. (n.d.). Retrieved March 03, 2016, from http://xenia.media.mit.edu/~gan/Gan/Education/NUS/Physics/MScThesis/Chapter1.html

2. Stokes, E. B., & Doering, E. (n.d.). LabView as a music synthesizer laboratory learning environment. Retrieved March 3, 2016.

3. Chowning, J. M. (1977). The Synthesis of Complex Audio Spectra by Means of Frequency Modulation. *Computer Music Journal, 1*(2), 46-54. Retrieved March 3, 2016.

4. Kleimola, J., Lazzarini, V., Valimaki, V., & Timoney, J. (2011). Feedback Amplitude Modulation Synthesis. *EURASIP Journal on Advances in Signal Processing*. Retrieved March 3, 2016.

5. Burk, P. Polansky, L. Repetoo, D. Roberts, M. Rockmore, D. 2011 “Music and Computers” Columbia University. Retrieved March 2, 2016. Web.

6. Kleczkowski, P. 1989 “Group Additive Synthesis” Computer Music Journal 13.1:12-20

7. Sasaki, L. H., Smith, K. C. 1980 “A Simple Data Reduction Scheme for Additive Synthesis” Computer Music Journal 4.1: 22-24

8. Laurson, M., Erkut, C., & Valimaki, V. (2000). Methods for Modeling Realistic Playing in Plucked-String Synthesis Analysis, Control and Synthesis. *Helsinki University of Technology*. Retrieved March 03, 2016.

9. Butterworth, S. (1930). On the Theory of Filter Amplifiers. *Experimental Wireless*. Retrieved March 3, 2016.

10. Wu, T. (2013, May 14). MW & RF Design: Filters. Retrieved March 3, 2016, from http://ntuemc.tw/upload/file/201102212235523ca26.pdf

11. Tan, B. G., & Lim, S. M. (n.d.). Automated Parameter Optimization for Double Frequency Modulation Synthesis.