BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF MECHANICAL ENGINEERING

FAKULTA STROJNÍHO INŽENÝRSTVÍ

INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

ÚSTAV AUTOMATIZACE A INFORMATIKY

ALGORITHMS FOR SYNTHESIS OF MUSICAL SOUNDS

ALGORITMY PRO SYNTÉZU HUDEBNÍCH ZVUKŮ

BACHELOR'S THESIS

BAKALÁŘSKÁ PRÁCE

AUTHOR AUTOR PRÁCE	Samuel Novák
SUPERVISOR VEDOUCÍ PRÁCE	Ing. Jan Roupec, Ph.D.

BRNO 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání bakalářské práce

Ústav:	Ústav automatizace a informatiky
Student:	**Samuel Novák**
Studijní program:	Strojírenství
Studijní obor:	Základy strojního inženýrství
Vedoucí práce:	**Ing. Jan Roupec, Ph.D.**
Akademický rok:	2017/18

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Algoritmy pro syntézu hudebních zvuků

Stručná charakteristika problematiky úkolu:

Cílem bakalářské práce je charakteristika problému syntézy hudebních zvuků, odvození fázového akumulátoru a implementace vybraných tvarovacích funkcí. V práci bude provedeno formální rozdělení problému na realizaci fázového akumulátoru a nalezení vhodné tvarovací funkce. Bude zkonstruován sinový oscilátor, provedena modifikace sinového oscilátoru za účelem získání vlny s bohatším harmonickým spektrem a odvozen pilový oscilátor s unisono efektem.

Cíle bakalářské práce:

Definice pojmů potřebných pro popis algoritmů oscilátorů z pohledu fázového zkreslení a skládání zobrazení.
Odvození fázového akumulátoru.
Implementace sinového oscilátoru.
Modifikace některého ze základních typů oscilátorů za účelem napodobení známého efektu či vytvoření nového efektu.

Seznam doporučené literatury:

KLEIMOLA, Jari, Victor LAZZARINI, Joseph TIMONEY a Vesa VÄLIMÄKI. Phaseshaping Oscillator Algorithms for Musical Sound Synthesis. Dostupné z: http://smcnetwork.org/files/proceedings/2010/15.pdf

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2017/18

V Brně, dne

L. S.

doc. Ing. Radomil Matoušek, Ph.D.
ředitel ústavu

doc. Ing. Jaroslav Katolický, Ph.D.
děkan fakulty

ABSTRAKT

Práce se zabývá problematikou syntézy hudebních zvuků, konktrétně sestavováním algoritmů oscilátorů. K problému je přistupováno z pohledu skládání funkcí a fázového zkreslení. Při návrhu potřebných funkcí bylo užito interpolačních polynomů, iteračních metod a optimalizace. Byl zkonstruován fázový akumulátor a sinová tvarovací funkce. Dále byl předveden způsob, jak zkreslit průběh sinové vlny. Nakonec byla popsána metoda pro syntézu rozladěných hlasů pilové vlny.

ABSTRACT

This work concerns the problem of synthesizing musical sounds, specifically the problem of designing oscillator algorithms. The problem is approached from the perspective of function composition and phase distortion. Interpolation polynomials, iteration methods and optimization were employed when designing the necessary functions. Phase accumulator and a sine waveshaping function were constructed. Furthermore, a way of distorting the sine waveform was presented. Finally, a method of synthesizing detuned voices of a sawtooth wave was described.

KLÍČOVÁ SLOVA

syntéza zvuku, algoritmy oscilátorů, fázové zkreslení, fázový akumulátor, tvarovací funkce, unisono efekt

KEYWORDS

sound synthesis, oscillator algorithms, phase distortion, phase accumulator, waveshaping function, unison effect

BIBLIOGRAFICKÁ CITACE

NOVÁK, Samuel. *Algorithms for Synthesis of Musical Sounds.* Brno, 2018. Bachelor's thesis. Brno University of Technology, Faculty of Mechanical Engineering, Institute of Automation and Computer Science.

ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že tato práce je mým původním dílem, zpracoval jsem ji samostatně pod vedením Ing. Jana Roupce, Ph.D. a s použitím literatury uvedené v seznamu literatury.

V Brně dne 25. 5. 2018 ……………………………………………….

Samuel Novák

Contents

1 Introduction

Musical sounds can be produced on the devices called sound synthesizers. Ordinary synthesizers consist of oscillators, filters and envelope generators. Oscillators produce a wave with a desired waveform and a given base frequency. Usually, the shape of the produced waveform is static, however oscillators which allow for the waveform to be continuously modulated do exist. Subsequently, filters are used to alter the harmonic content of the produced waveveform. Finally, envelope generators introduce a change in the intensity of the sound over time, typically in response to note keypress events.

This work concerns oscillators only. In the first segment, basic terminology for describing oscillator algorithms in terms of function composition and phase distortion is introduced. The problem is formally divided into assembling a phase accumulator and obtaining a suitable waveshaping function. The construction of the phase accumulator is performed step by step. Finally, a way of approximating a sine wave is demonstrated and a sine waveshaping function is specified.

In the second segment, novel oscillator algorithms are presented. Self-modulation of a sine function is investigated in order to produce a distorted sine waveform. Subsequently, a particular phase distorting function for emulating the obtained effect is devised. Later, the interference of slighly detuned sawtooth waves is investigated. Obtained findings serve as a basis for an alternative method of producing a sawtooth wave unison effect.

The aim of this work is to both provide a brief introduction into the matter of sound synthesis and to present novel findings. Intentionally, this work begins with the basic problems and gradually approaches the more difficult ones.

The complex oscillators presented in the second segment can be used as an alternative to the basic oscillators found in traditional sound synthesizers. Furthermore, they can be used in phase distortion, phase modulation and frequency modulation settings in order to obtain complex sounds more easily.

It would be appropriate to highlight the paper [1], which initially motivated the creation of this work. Without being inspired by the paper, this work wouldn't embrace the concepts of function composition and phase distortion to such an extent.

2 Continuous Time Periodic Signals

When synthesizing musical sounds a common problem we need to solve is that of generating a periodic signal of a particular base frequency and a desired waveform. In discrete time systems this is often accomplished by generating a primitive signal of the particular base frequency and subsequently applying a suitable function onto such signal in order to obtain the desired waveform. Before we reach the solution for discrete time systems, let's briefly delve into the matter of continuous time periodic signals and their representation.

To illustrate the problem, let's assume we want to obtain a sinusoidal signal. Such signal can be represented symbolically by the following equation:

$$y(t) = sin(2\pi ft) \tag{1}$$

where $y(t)$ is the output signal, f denotes the base frequency of the signal and t stands for time. Even though in this particular case we consider f a parameter, generally it may be a function of time. If we define $\varphi(t) = ft$ we can rewrite Equation 1 in the following way:

$$y(t) = sin(2\pi\varphi(t)) \tag{2}$$

where $\varphi(t)$ represents the phase signal. If we also define $g(x) = sin(2\pi x)$ we can express Equation 2 in terms of function composition:

$$y(t) = (g \circ \varphi)(t) \tag{3}$$

where g is the waveshaping function. Furthermore, Equation 3 can be visualized using a block diagram as depicted in Figure 1.

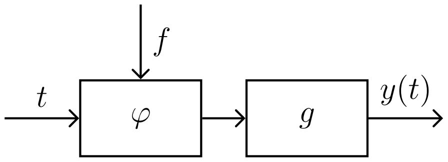

Figure 1: Block diagram corresponding to Equation 3.

As apparent from Figure 1 the waveshaping function g is being applied on the phase signal $\varphi(t)$ in order to yield the output signal $y(t)$. However, a different point of view is also possible. We can imagine the phase signal $\varphi(t)$ reading out corresponding values from the waveshaping function g. This idea becomes of importance when we synthesize musical sounds digitally. In such instance, the waveshaping function g may be an interpolated wavetable.

Let's further investigate the relationship of the phase signal $\varphi(t)$ and the waveshaping function g by visually comparing the phase signal $\varphi(t)$ with the output signal $(g \circ \varphi)(t)$ as depicted in Figure 2.

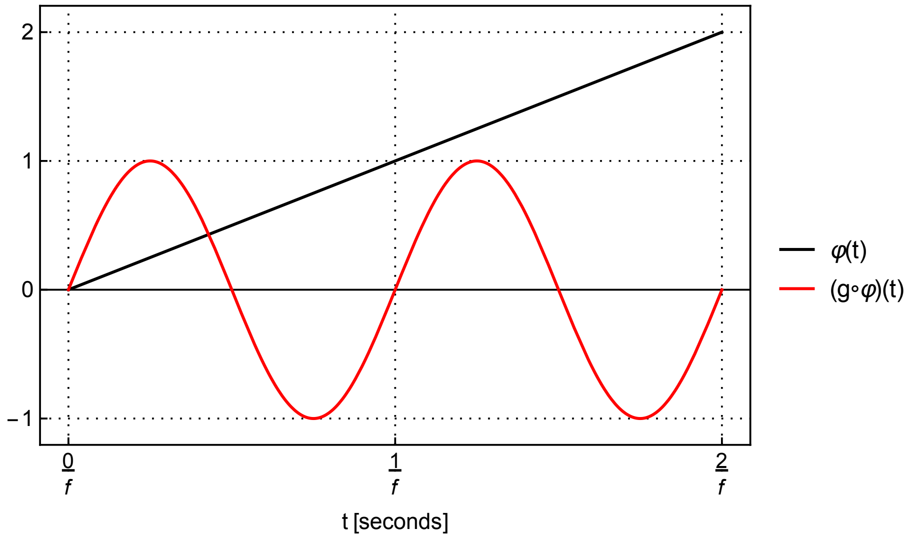

Figure 2: Graph comparing the phase signal $\varphi(t)$ with the output signal $(g \circ \varphi)(t)$.

As could be observed, the output signal $(g \circ \varphi)(t)$ is periodic even though the phase signal $\varphi(t)$ is aperiodic. Consequently, the waveshaping function g must be periodic with period 1. Since the codomain of the phase signal $\varphi(t)$ is $\mathbb{R}$, the domain of the waveshaping function g must also be $\mathbb{R}$.

Now comes the time to ask ourselves if the phase signal $\varphi(t)$ could be modified to be periodic without affecting the output signal $(g \circ \varphi)(t)$? Since the phase signal $\varphi(t)$ is linear, perhaps a suitable modification could be piecewise linear. Let's introduce a boundary function g_b with the following definition:

$$g_b(x) = x - floor(x) = x \bmod 1 \tag{4}$$

Furthermore, let's propose a modified phase signal $\varphi_b(t)$ defined as follows:

$$\varphi_b(t) = (g_b \circ \varphi)(t) \tag{5}$$

By visually comparing the phase signal $\varphi(t)$ with the modified phase signal $\varphi_b(t)$, it could be verified that the proposed modification indeed produces a piecewise linear periodic signal as depicted in Figure 3.

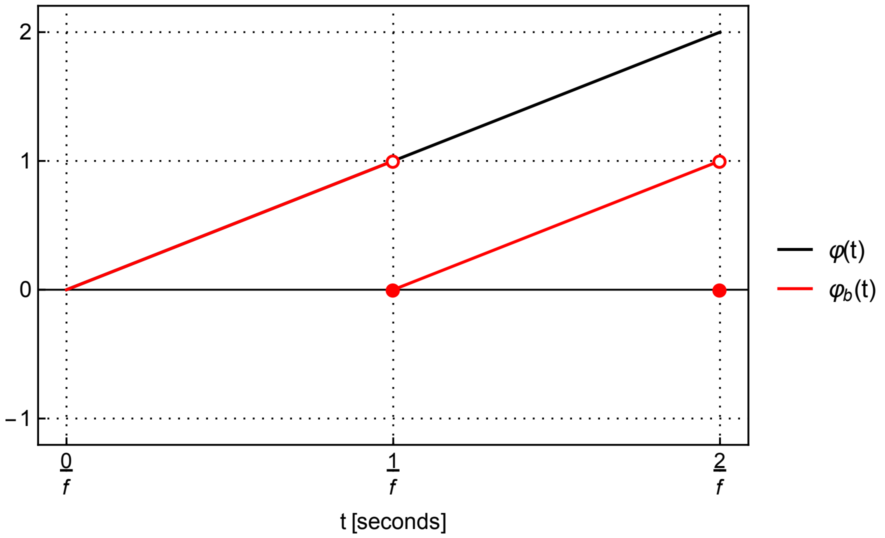

Figure 3: Graph comparing the phase signal $\varphi(t)$ with the modified phase signal $\varphi_b(t)$.

Finally, let's compare the modified phase signal $\varphi_b(t)$ with the output signal $(g \circ \varphi_b)(t)$ obtained using the modified phase signal $\varphi_b(t)$ as depicted in Figure 4.

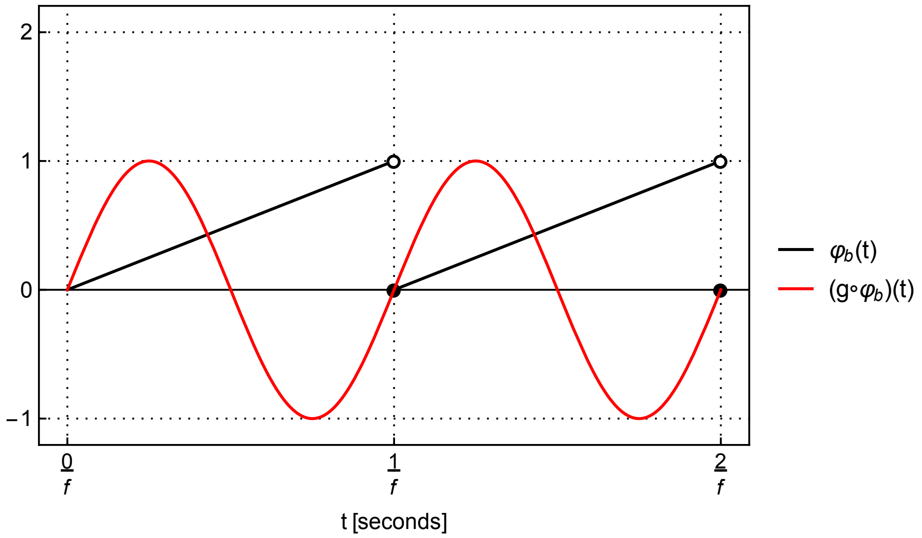

Figure 4: Graph comparing the modified phase signal $\varphi_b(t)$ with the output signal $(g \circ \varphi_b)(t)$.

It is possible to conclude that the output signal wasn't affected by modifying the phase signal $\varphi(t)$. The periodic nature of the output signal $(g \circ \varphi_b)(t)$ is now satisfied by the modified phase signal $\varphi_b(t)$ itself being periodic, thus the waveshaping function g no longer needs to be periodic. Furthermore, since the codomain of the modified phase signal $\varphi_b(t)$ is $\langle 0, 1)$, the domain of the waveshaping function g also becomes $\langle 0, 1)$ rather than $\mathbb{R}$. Finally, it is now feasible to represent the values of the phase signal using floating-point numbers.

3 Phase Accumulator

Let's shift our focus to discrete time systems and try to reproduce the modified phase signal $\varphi_b(t)$ in such setting. Since the modified phase signal $\varphi_b(t)$ is piecewise linear, it should be possible to recreate the linear segments by accumulating properly sized phase increments on each successive tick of the sample rate clock.

First, let's determine the size of the phase increment. We will consider the base frequency f to be a function of the sample number and denote it as $f(n)$, where n stands for the sample number. Also, let's consider k to be the number of samples per cycle, which can be calculated as follows:

$$k = T(n) \cdot f_s \tag{6}$$

where $T(n)$ stands for the base period, which is the reciprocal of $f(n)$, and f_s denotes the sampling frequency. In order to proceed further, let's consider i to be the size of the phase increment. Assuming we keep the base frequency $f(n)$ constant, the phase signal must add up to 1 during the cycle, thus the size of the phase increment i can be derived in the following way:

$$i = \frac{1}{k} = \frac{1}{T(n) \cdot f_s} = \frac{f(n)}{f_s} \tag{7}$$

For the sake of convenience, let's also define an increment function g_{inc} as follows:

$$g_{inc}(f, f_s) = \frac{f}{f_s} \tag{8}$$

Let's continue by proposing a scheme that calculates a sum of the current value of the phase increment and the previous value of the phase signal and passes the sum into the boundary function g_b in order to obtain the current value of the phase signal. Such scheme describes a phase accumulator, which can be represented by the block diagram shown in Figure 5.

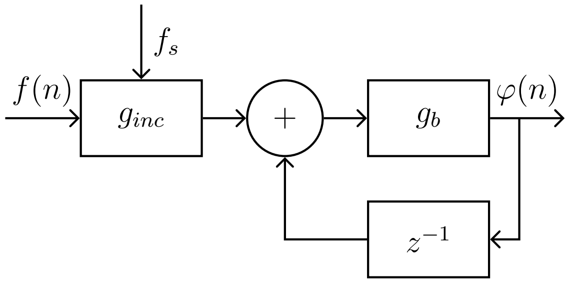

Figure 5: Block diagram representing the phase accumulator.

Note that z^{-1} denotes a unit delay and $\varphi(n)$ stands for the current value of the phase signal. If we choose $\varphi(0)$ to be a predetermined initial phase, it becomes possible to calculate the successive values of the phase signal in an iterative fashion using the following formula:

$$\varphi(n) = g_b\Big(g_{inc}\big(f(n), f_s\big) + \varphi\big(n-1\big)\Big) \tag{9}$$

By applying a waveshaping function g on the discrete time phase signal $\varphi(n)$, we obtain an oscillator $(g \circ \varphi)(n)$ in accordance with the formal distinction outlined in the previous chapter. Also, it is possible to insert a phase distorting function between the phase signal $\varphi(n)$ and the waveshaping function g in order to perform phase distortion synthesis. This work will further be devoted to constructing suitable waveshaping and phase distorting functions.

Since the discrete time phase signal $\varphi(n)$ reads out appropriate values from the waveshaping function g, the time discretization is ensured solely by the discrete time phase signal $\varphi(n)$, which makes it possible to handle the waveshaping function g as if the setting was continuous time.

For the sake of completeness, we shall note that the implementation of a phase accumulator we described permits the base frequency $f(n)$ to be negative, which allows for a through-zero frequency modulation to be performed.

4 Sine Wave Oscillator

The first waveshaping function we are going to create is that belonging to a sine wave oscillator, which is commonly found in both analog and digital synthesizers. In order to approximate a sine wave, it is sufficient to approximate the function in the first quadrant only. The values from the remaining quadrants are to be resolved by altering the way of reading out the values from the approximation and by changing the sign.

For the sake of convenience, let's define a function f in a way that the values $x \in \langle 0, 1 \rangle$ align with the first quadrant:

$$f(x) = sin(\frac{\pi}{2} \cdot x) \tag{10}$$

Such function would typically be approximated using a Taylor polynomial. However, the method we are about to demonstrate works by constructing an interpolation polynomial between the endpoints of the segment being approximated. By properly prescribing first order derivatives at the endpoints, smoothness of the resulting waveshaping function can be ensured.

Let's introduce a shaping function g_{sh} and prescribe the following properties:

$$\begin{aligned} g_{sh}(0) &= 0 \\ g'_{sh}(0) &= \frac{\pi}{2} \\ g''_{sh}(0) &= 0 \\ g_{sh}(1) &= 1 \\ g'_{sh}(1) &= 0 \end{aligned} \tag{11}$$

By solving the corresponding system of linear equations the shaping function g_{sh} can be obtained in the following form:

$$g_{sh}(x) = (-3 + \pi) \cdot x^4 + \frac{8 - 3\pi}{2} \cdot x^3 + \frac{\pi}{2} \cdot x \tag{12}$$

The accuracy of the approximation can be verified by visually comparing the approximated function f with the shaping function g_{sh} as shown in Figure 6.

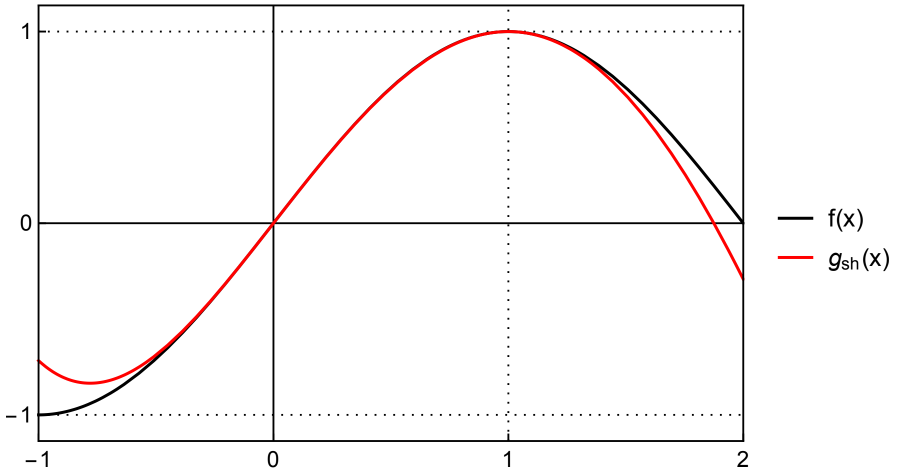

Figure 6: Graph comparing the approximated function f with the shaping function g_{sh}.

In order to proceed, we should notice that the half-periods of a sine wave differ solely by their signs. Therefore it should be possible to generate both half-periods using the same function, assuming the sign is subsequently changed if necessary. Let's consider a half-period function g_{hp}, which creates an individual unit ramp for each of the half-periods, defined as follows:

$$g_{hp}(x) = \begin{cases} 2x, & \text{if } x < 0.5 \\ 2x - 1, & \text{if } x \geq 0.5 \end{cases} \tag{13}$$

Additionally, let's introduce a sign function g_{sgn} defined in the following way:

$$g_{sgn}(x) = \begin{cases} 1, & \text{if } x < 0.5 \\ -1, & \text{if } x \geq 0.5 \end{cases} \tag{14}$$

Both functions can be compared visually in Figure 7.

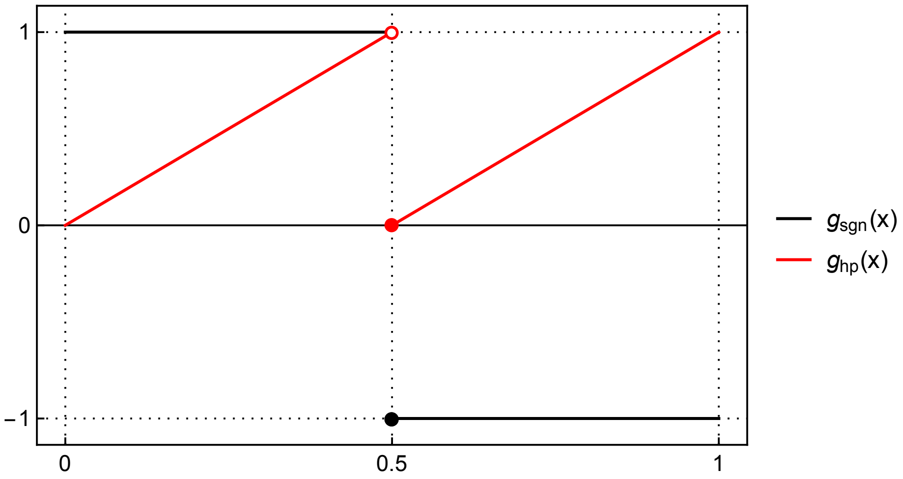

Figure 7: Graph comparing the sign function g_{sgn} with the half-period function g_{hp}.

It should be possible to properly read out the half-periods from the shaping function g_{sh} if we start reading out backwards when we approach the second quadrant. In order to accomplish this, let's propose a folding function g_{fold}, which folds a unit ramp into a triangle, defined as follows:

$$g_{fold}(x) = \begin{cases} 2x, & \text{if } x < 0.5 \\ 2 - 2x, & \text{if } x \geq 0.5 \end{cases} \tag{15}$$

The effect of applying the folding function g_{fold} on the half-period function g_{hp} can be observed in Figure 8.

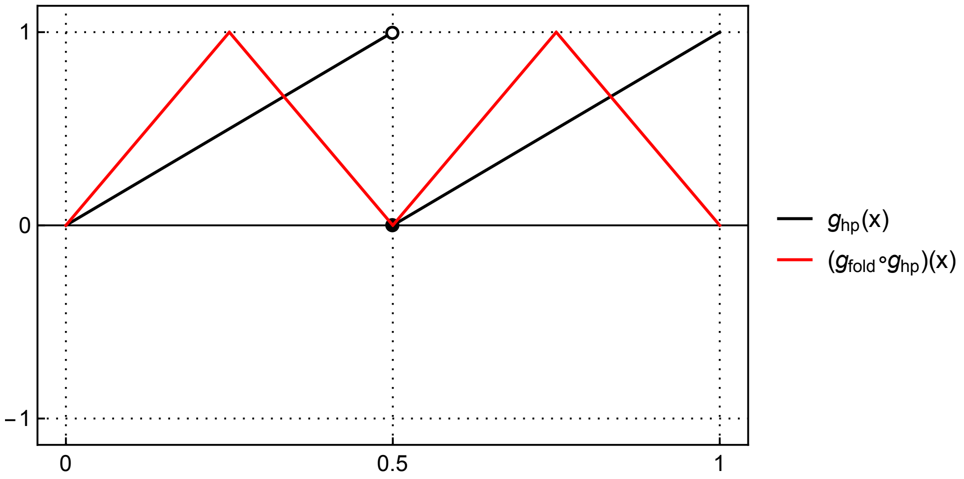

Figure 8: Graph depicting the effect of applying the folding function g_{fold} on the half-period function g_{hp}.

Further progress can be made by applying the shaping function g_{sh}. In order to merge the three operations, let's introduce an unipolar function g_{uni} defined in the following way:

$$g_{uni}(x) = (g_{sh} \circ g_{fold} \circ g_{hp})(x) \tag{16}$$

It can be verified visually that the unipolar function g_{uni} indeed produces the half-periods in Figure 9.

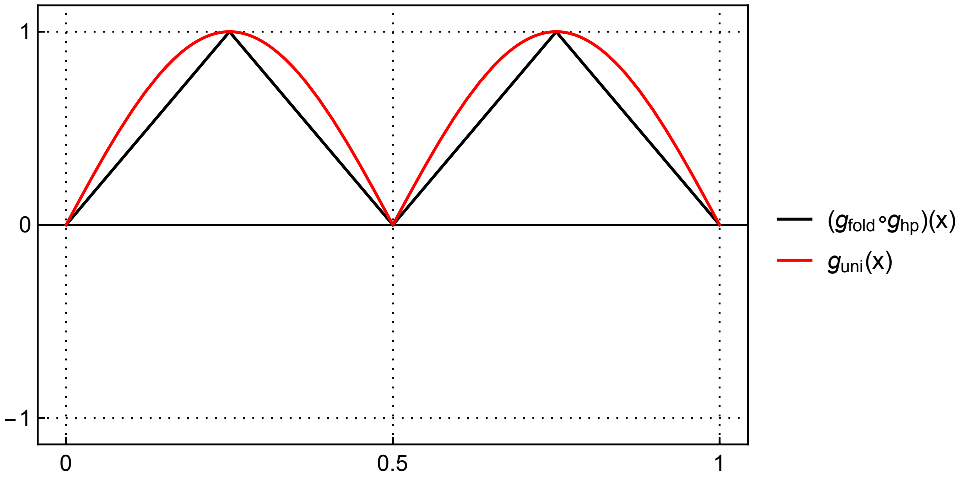

Figure 9: Graph comparing the unipolar function g_{uni} with the folding function g_{fold} applied on the half-period function g_{hp}.

Finally, the sign function g_{sgn} as defined previously can be employed in order to obtain a sine waveshaping function g_{sin} defined as follows:

$$g_{sin}(x) = g_{sgn}(x) \cdot g_{uni}(x) \tag{17}$$

The effect of multiplying the unipolar function g_{uni} by the sign function g_{sgn} can be observed in Figure 10.

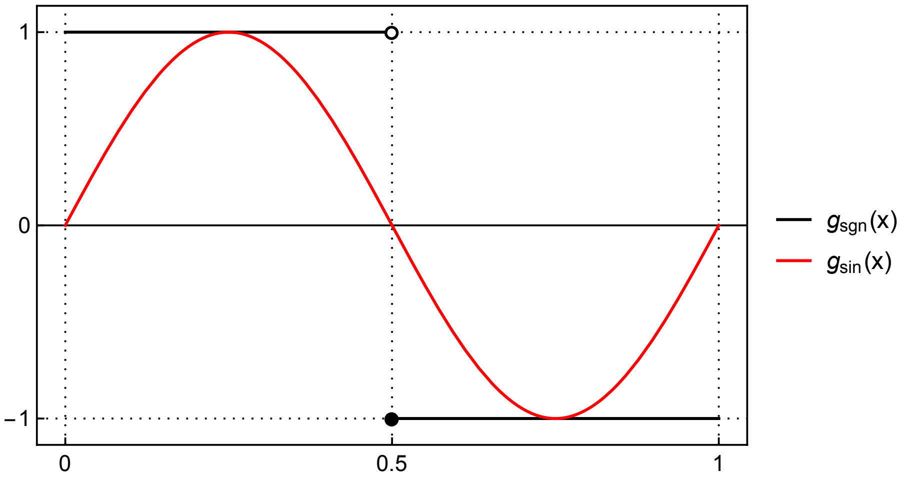

Figure 10: Graph comparing the sign function g_{sgn} with the function g_{sin}.

Additionally, it is possible to represent the sine waveshaping function g_{sin} using a block diagram as shown in Figure 11.

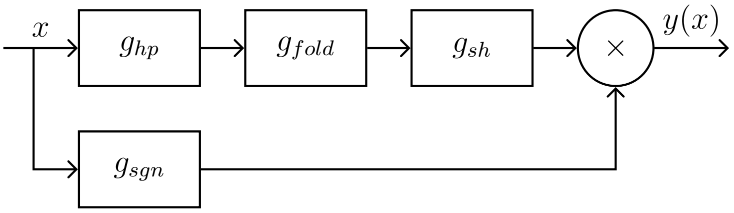

Figure 11: Block diagram representing the sine waveshaping function g_{sin}.

In order to evaluate the quality of the approximation, harmonic analysis of the sine waveshaping function g_{sin} could be performed as shown in Table 1. Notice that the approximation error manifests itself primarily in the form of parasitic higher-order harmonic components.

	Amplitude	
Harmonic	Linear	dBFS
1	0.998506	-0.01
3	0.000629	-64.03
5	0.000688	-63.25
7	0.000078	-82.20
9	0.000056	-85.05
11	0.000015	-96.34
13	0.000012	-98.42
15	0.000005	-106.47

Table 1: The first 16 harmonic components of the sine waveshaping function g_{sin}. Only non-zero components appear in the table.

By subtracting the amplitude of the most prominent parasitic component, which happens to be the 5th harmonic component, from the amplitude of the fundamental component a signal-to-noise ratio equal to 63.24 dB can be obtained, which should be deemed sufficient considering practical listening circumstances.

5 Sine Wave Distortion Effect

With a sine waveshaping function at our disposal, the question that arises is whether a way of distorting the phase signal reading out from the sine waveshaping function that would manifest itself in altering the resulting waveform in a controlled manner exists. Such alteration of the waveform could be performed in order to obtain a richer harmonic spectrum.

One way of distorting the phase signal can be found through a self-modulation of a sine function. Let's consider the scheme depicted in Figure 12, where g_{nl} represents an arbitrary nonlinear function and m stands for a modulation index.

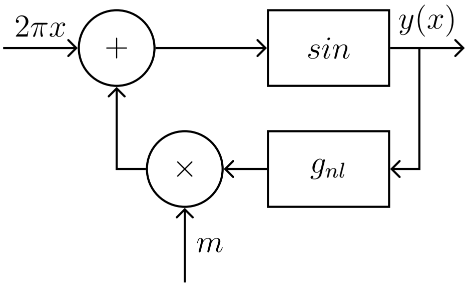

Figure 12: Block diagram depicting the self-modulation of a sine function.

As evident from the block diagram, the output of the sine function sin is fed back as an input after being transformed by the nonlinear function g_{nl}. The amount of feedback is controlled by the modulation index m. It is possible to express the output signal $y(x)$ in the form of an implicit function as follows:

$$y(x) = sin\big(2\pi x + m \cdot (g_{nl} \circ y)(x)\big) \tag{18}$$

In order to focus on a particular case, let's define the nonlinear function g_{nl} in the following way:

$$g_{nl}(x) = (cos \circ arcsin)(x) - cos(0) = \sqrt{1 - x^2} - 1 \tag{19}$$

Consequently, a function f obtained by substituting for the nonlinear function g_{nl} in Equation 18 can be introduced:

$$f(x, m) = sin\big(2\pi x + m \cdot \sqrt{1 - f(x, m)^2} - m\big) \tag{20}$$

The different waveforms generated by the function f as we change the modulation index m in the range $\langle 0, 1\rangle$ can be observed in Figure 13.

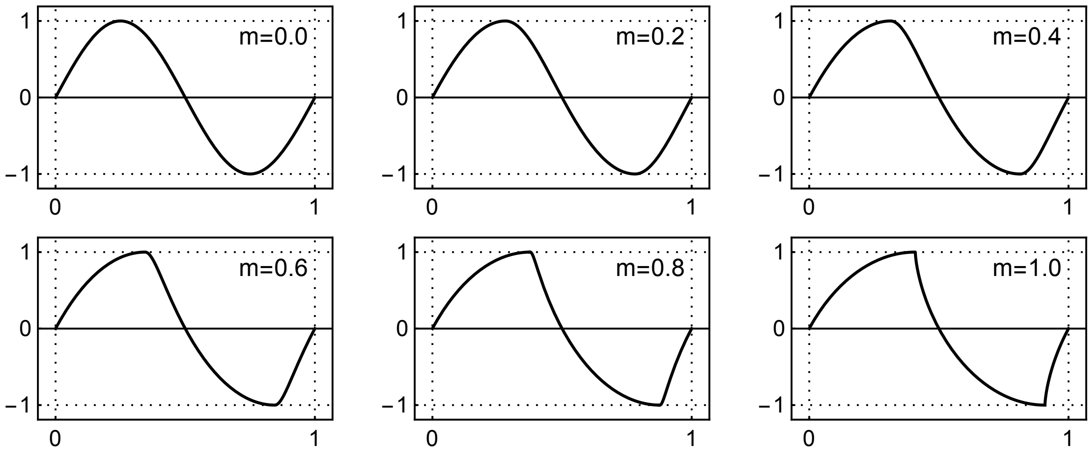

Figure 13: Graphs depicting the influence of the modulation index m on the waveform generated by the function f.

As apparent from Figure 13, the resulting waveform morphs from a sine wave to a waveform resembling sawtooth/triangle wave as the modulation index m increases. In other words, the modulation index m mimicks the effect of a low-pass filter as found in a traditional sound synthesizer. This property makes the function f interesting from a musical perspective.

Since the function f is implicit, efforts on approximating the function typically lead to iterative methods. In this case, approximating the function f would require too many iterations. To overcome the problem, we will propose a function for generating a reasonable initial guess. Later, a way of achieving nearly optimal relaxation will be introduced. In the end, it should be possible to obtain a satisfactory approximation using 2 iterations only.

Since the half-periods of the approximated function f are symmetric, we will initially focus on approximating the first half-period. For the sake of convenience, let's introduce a half-period function f_{hp} defined as follows:

$$f_{hp}(x, m) = f\big(\frac{x}{2}, m\big) \tag{21}$$

If we continue by solving the equation $f_{hp}(x_{div}, m) = 1$, a value x_{div} that divides the half-period in two distinctive segments can be obtained. Let's define a division function g_{div}, which yields the value x_{div}, in the following way:

$$g_{div}(m) = \frac{m}{\pi} + \frac{1}{2} \tag{22}$$

The segments can be visualized by plotting the half-period function f_{hp} with the value x_{div} indicated as shown in Figure 14.

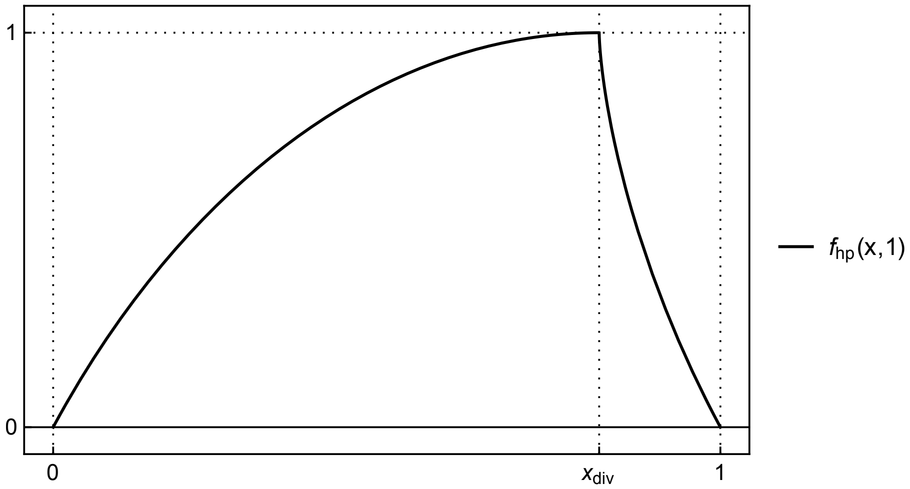

Figure 14: Graph of the half-period function f_{hp}.

In order to proceed, an interpolation polynomial for each of the segments will be proposed. Let's denote the first interpolation polynomal h_1 and prescribe the following properties:

$$\begin{aligned} h_1(0) &= 0 \\ h_1'(0) &= \pi \\ h_1(x_{div}) &= 1 \\ h_1'(x_{div}) &= 0 \end{aligned} \tag{23}$$

By solving the corresponding system of linear equations the first interpolation polynomal h_1 can be obtained in the following form:

$$h_1(x, x_{div}) = \frac{-2 + \pi x_{div}}{x_{div}^3} \cdot x^3 + \frac{3 - 2\pi x_{div}}{x_{div}^2} \cdot x^2 + \pi \cdot x \tag{24}$$

In the same vein, let's denote the second interpolation polynomal h_2 and prescribe the following properties:

$$\begin{aligned} h_2(0) &= 1 \\ h_2'(0) &= s \\ h_2(l) &= 0 \\ h_2'(l) &= -\pi \end{aligned} \tag{25}$$

where l denotes the segment length and s stands for the slope. By solving the corresponding system of linear equations the second interpolation polynomal h_2 can be obtained in the following form:

$$h_2(x, l, s) = \frac{2 + l(-\pi + s)}{l^3} \cdot x^3 + \frac{-3 + l(\pi - 2s)}{l^2} \cdot x^2 + s \cdot x + 1 \tag{26}$$

Both interpolation polynomials are used inside an initializing function g_{init}, which computes the initial guess, defined as follows:

$$g_{init}(x, x_{div}, s) = \begin{cases} h_1(x, x_{div}), & \text{if } x < x_{div} \\ h_2(x - x_{div}, 1 - x_{div}, s), & \text{if } x \geq x_{div} \end{cases} \tag{27}$$

In order to determine the slope s, the following optimization problem will be solved for chosen values of the modulation index m:

$$\min_{s} \int_{g_{div}(m)}^{1} \Big(g_{init}\big(x, g_{div}(m), s\big) - f_{hp}\big(x, m\big)\Big)^2 \, dx \tag{28}$$

Subsequently, a function in the form $a \cdot x^4$ is fitted to the points obtained by solving the optimization problem. This way, a slope function g_s can be determined as follows:

$$g_s(m) = -10.6736 \cdot m^4 \tag{29}$$

The slope function g_s can be visually compared with the points obtained by solving the optimization problem in Figure 15.

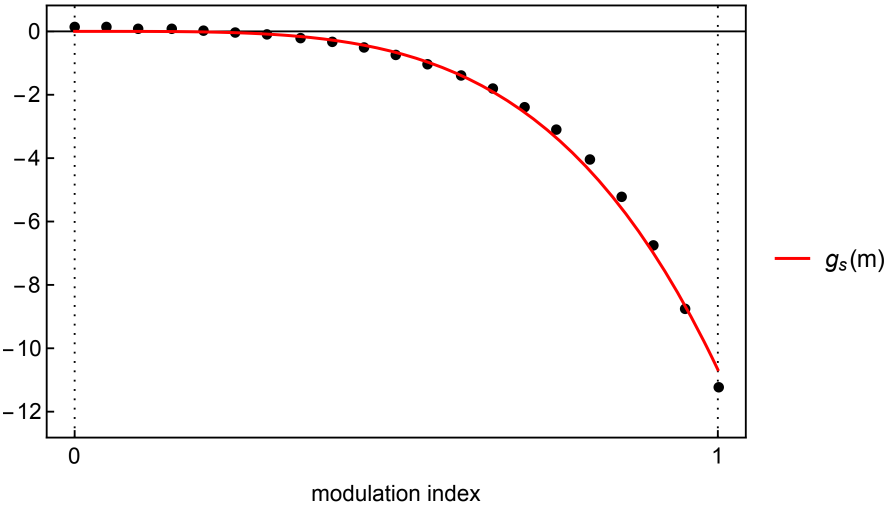

Figure 15: Graph comparing the slope function g_s (drawn red) with the points obtained by solving Equation 28 (drawn black).

Finally, it is possible to define the initial guess y_{init} in the following way:

$$y_{init}(x, m) = g_{init}\big(x, g_{div}(m), g_s(m)\big) \tag{30}$$

Furthermore, the initial guess y_{init} can be visually compared with the half-period function f_{hp} being approximated in Figure 16.

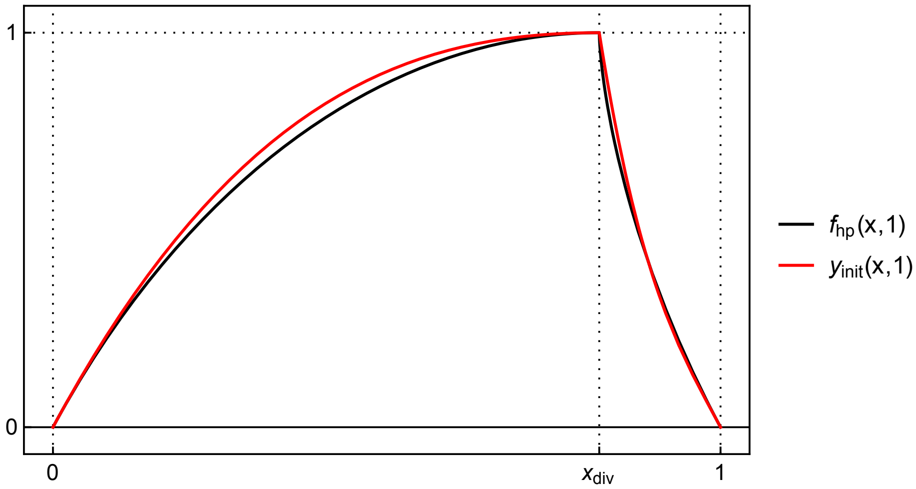

Figure 16: Graph comparing the initial guess y_{init} with the half-period function f_{hp}.

In order to progress towards computing an interation, let's introduce a phase function g_{ph} defined as follows:

$$g_{ph}(x,y,m) = \begin{cases} \dfrac{x}{2} + \dfrac{m \cdot \sqrt{1-y^2} - m}{2\pi}, & \text{if } y < 1 \\ \dfrac{x}{2} - \dfrac{m}{2\pi}, & \text{if } y \geq 1 \end{cases} \tag{31}$$

where y is the current estimate. Note that the definition of the phase function g_{ph} allows for the current estimate y to overshoot. Now it becomes possible to compute the first iteration y_{iter} in the following way:

$$y_{iter}(x,m) = (g_{sin} \circ g_{ph})\big(x, y_{init}(x,m), m\big) \tag{32}$$

With the first iteration y_{iter} at our disposal, it becomes possible to perform relaxation. Let's consider an approximate value y_{approx} defined as follows:

$$y_{approx}(x,m,\omega) = y_{init}(x,m) + \omega \cdot \big(y_{iter}(x,m) - y_{init}(x,m)\big) \tag{33}$$

where ω denotes the relaxation factor. In order to determine the relaxation factor ω, the optimal relaxation factor will be computed for chosen values of the modulation index m in each segment separately. In the first segment, the following optimization problem will be solved:

$$\min_{\omega} \int_{0}^{g_{div}(m)} \Big((g_{sin} \circ g_{ph})\big(x, y_{approx}(x,m,\omega), m\big) - f_{hp}(x,m)\Big)^2 \, dx \tag{34}$$

In the second segment, the optimization problem is formulated similarly:

$$\min_{\omega} \int_{g_{div}(m)}^{1} \Big((g_{sin} \circ g_{ph})\big(x, y_{approx}(x,m,\omega), m\big) - f_{hp}(x,m)\Big)^2 \, dx \tag{35}$$

By fitting a function in the form $a \cdot x^2 + b \cdot x + 1$ to the points obtained by solving the optimization problem in the first segment, we obtain a polymonial h_3 as follows:

$$h_3(m) = 0.202709 \cdot m^2 - 0.665161 \cdot m + 1 \tag{36}$$

The polymonial h_3 can be visually compared with the points obtained by solving the optimization problem in the first segment in Figure 17.

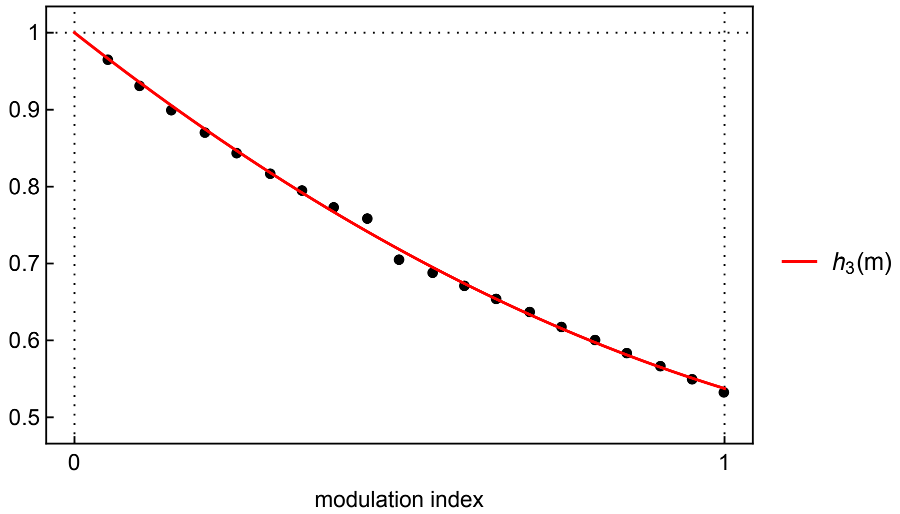

Figure 17: Graph comparing the polymonial h_3 (drawn red) with the points obtained by solving Equation 34 (drawn black).

In the same vein, by fitting a function in the form $a \cdot x^3 + 1$ to the points obtained by solving the optimization problem in the second segment, a polymonial h_4 can be obtained as follows:

$$h_4(m) = 3.39188 \cdot m^3 + 1 \tag{37}$$

The polymonial h_4 can be visually compared with the points obtained by solving the optimization problem in the second segment in Figure 18.

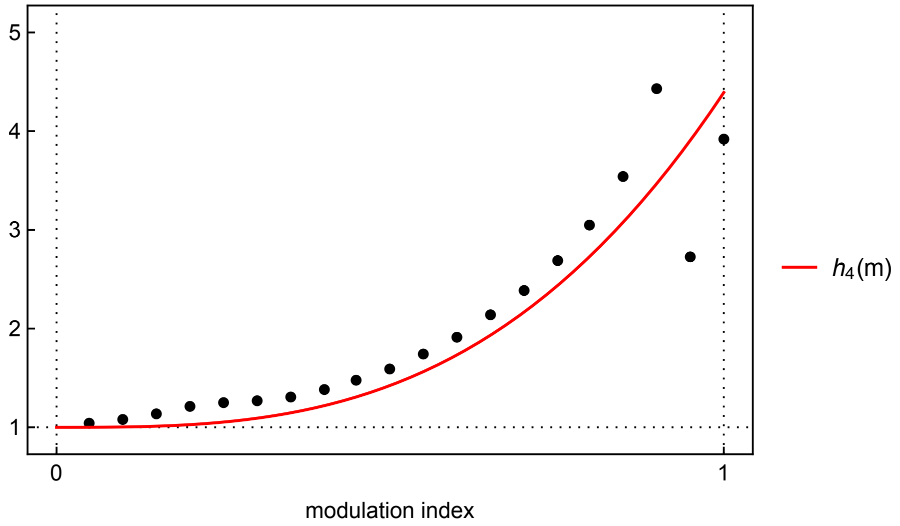

Figure 18: Graph comparing the polymonial h_4 (drawn red) with the points obtained by solving Equation 35 (drawn black).

Finally, it becomes possible to introduce a coefficient function g_{coef} defined in the following way:

$$g_{coef}(x, x_{div}, m) = \begin{cases} h_3(m), & \text{if } x < x_{div} \\ h_4(m), & \text{if } x \geq x_{div} \end{cases} \tag{38}$$

Subsequently, the relaxation factor ω can be determined as follows:

$$\omega(x, m) = g_{coef}\big(x, g_{div}(m), m\big) \tag{39}$$

With the relaxation factor ω at our disposal, we can proceed by contructing an approximation function g_{approx}, which yields the final approximation of the phase signal for the first half-period, defined in the following way:

$$g_{approx}(x, m) = g_{ph}\Big(x, y_{approx}\big(x, m, \omega(x, m)\big), m\Big) \tag{40}$$

This operation represents the second iteration. By applying the sine waveshaping function g_{sin} on the approximation function g_{approx}, it becomes possible to visually evaluate the quality of the approximation by comparison with the half-period function f_{hp} as shown in Figure 19.

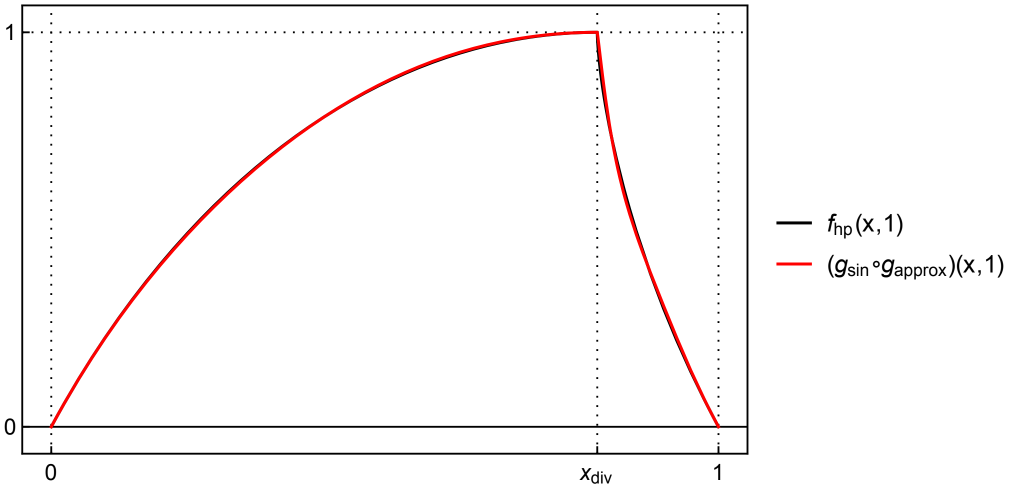

Figure 19: Graph comparing the sine waveshaping function g_{sin} applied on the approximation function g_{approx} with the half-period function f_{hp} being approximated.

In order to proceed, the phase signal for the whole period must be reconstructed. Let's consider a shift function g_{shift} defined as follows:

$$g_{shift}(x) = \begin{cases} 0, & \text{if } x < 0.5 \\ 0.5, & \text{if } x \geq 0.5 \end{cases} \tag{41}$$

Assuming the phase signal for the second half-period can be obtained by adding a constant to the phase signal from the first half-period, it should be possible to reconstruct the phase signal for the whole period as outlied in Figure 20.

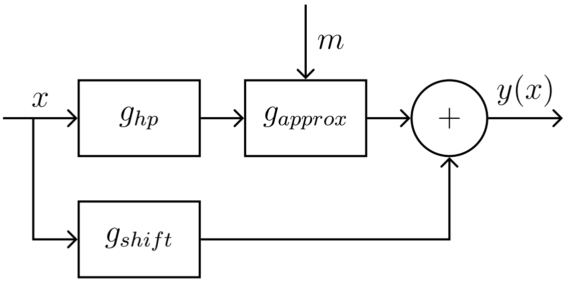

Figure 20: Block diagram representing the distortion function g_{dist}.

In accordance with Figure 20, we can introduce a distortion function g_{dist} defined in the following way:

$$g_{dist}(x,m) = g_{approx}\big(g_{hp}(x), m\big) + g_{shift}\big(x\big) \tag{42}$$

Note that g_{hp} refers to the half-period function defined in Equation 13. It can be visually verified that the distortion function g_{dist} indeed reads out the resulting waveform from the sine waveshaping function g_{sin} properly in Figure 21.

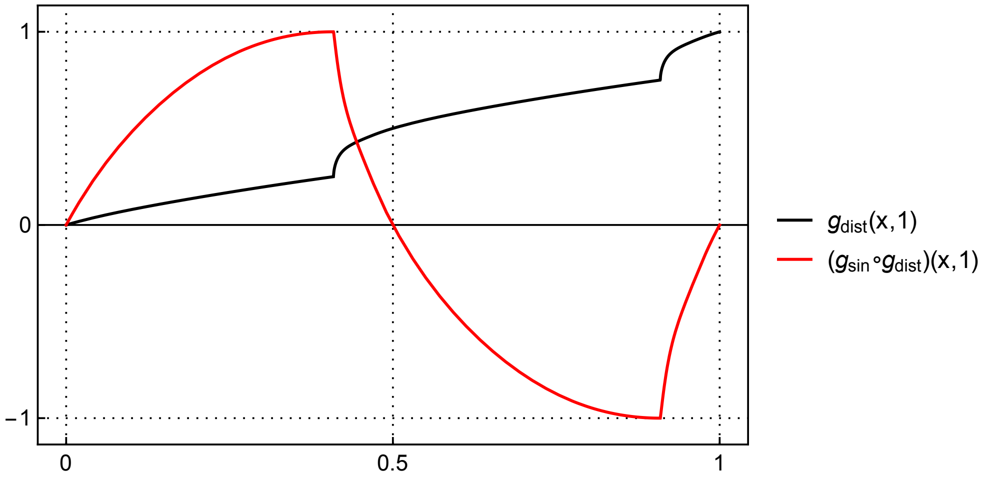

Figure 21: Graph depicting the distortion function g_{dist} reading out the resulting waveform from the sine waveshaping function g_{sin}.

Finally, the quality of the approximation can be inspected by comparing the composed function $(g_{sin} \circ g_{dist})$ with the function f being approximated for varying values of the modulation index m as shown in Figure 22.

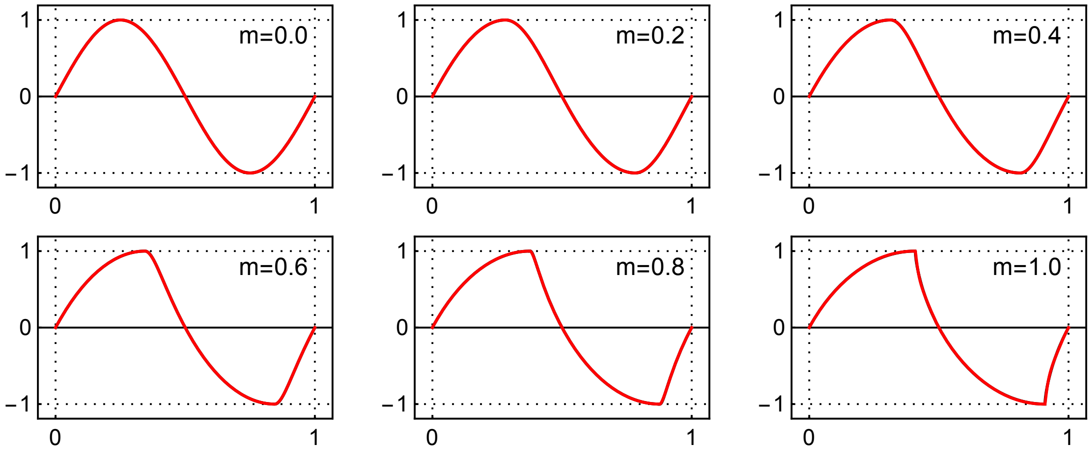

Figure 22: Graphs depicting the composed function ($g_{sin} \circ g_{dist}$) (drawn red) laid over the the function f (drawn black) for varying values of the modulation index m.

It should be noted that the distortion function g_{dist} can be used with any waveshaping function. Furthermore, it could be used alongside other functions distorting the phase. For example, let's consider a reverse function g_{rev} defined as follows:

$$g_{rev}(x) = \begin{cases} 0.5 - x, & \text{if } x < 0.5 \\ x, & \text{if } x \geq 0.5 \end{cases} \tag{43}$$

If we distort the phase signal using the reverse function g_{rev} before reaching the distortion function g_{dist}, it is possible to obtain a waveform resembling a sawtooth wave as depicted in Figure 23.

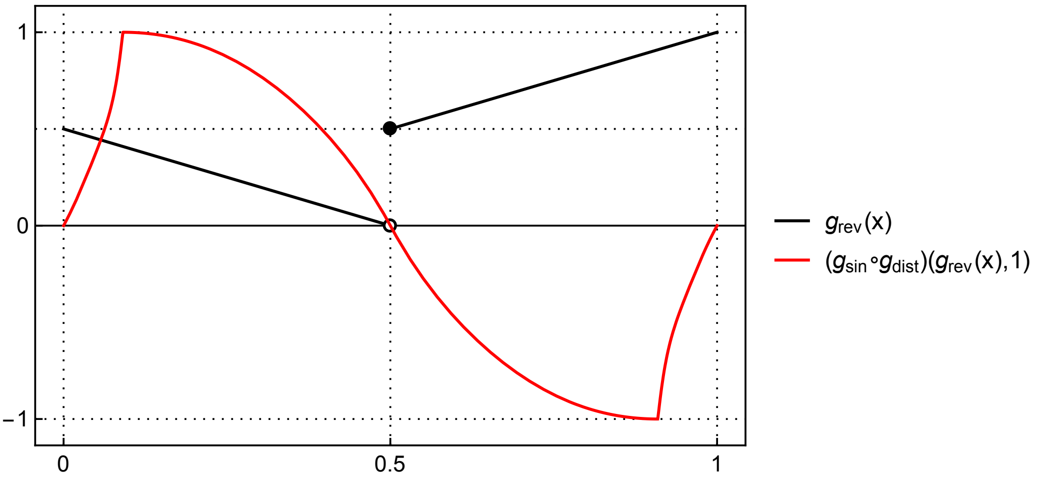

Figure 23: Graph depicting the reverse function g_{rev} and the waveform obtained using the reverse function g_{rev}.

6 Sawtooth Wave Unison Effect

In this chapter, the effect emerging from the interference of slightly detuned sawtooth waves will be examined and a method for emulating the effect will be devised. Slightly detuned waves are often combined in order to create a musically interesting change in timbre. This way, a time-variant timbre can be obtained from a combination of otherwise static sounds. The effect resulting from the combinaton of slightly detuned waves of the same waveform is usually called unison effect.

With regards to the particular case of combining slightly detuned sawtooth waves, a well-known instance of the effect can be found in the virtual analog synthesizer Roland JP-8000. This synthesizer featured a special oscillator called Supersaw, which emulated the sound of seven slightly detuned sawtooth waves combined. The resulting effect and its subtleties were extensively described in [2], including an emulation of the effect. Rather than emulating the particular effect found in Roland JP-8000, this work concerns obtaining a similar effect for use in the context of phase distortion synthesis.

Generally, synthesizing detuned voices requires introducing a separate phase accumulator for each of the voices. However, an algorithm for synthesizing a sawtooth wave with an arbitrary number of detuned sidebands using two phase accumulators only will be explained.

Assume a detuned sideband consisting of two sawtooth waves with the relative frequencies $(1 - d)$ and $(1 + d)$. This scenario can be described using the following function:

$$f_1(x) = \frac{1}{2} \cdot \big((1-d) \cdot x \bmod 1 + (1+d) \cdot x \bmod 1\big) \tag{44}$$

where d stands for the relative spread. With $d = 0.001$, the wavefroms generated by the function f_1 are shown in Figure 24. Notice that the individual snapshots align with a cycle of a sawtooth wave with a unit frequency. The interference of the sideband costituents can be perceived as a continous variation in the waveshape. Furthermore, the resulting variation in the waveshape is periodic, with the period determined as follows:

$$T_{side} = \frac{1}{d} \cdot T_{base} \tag{45}$$

where T_{base} represents the base period, which corresponds to a unit frequency in the relative terms.

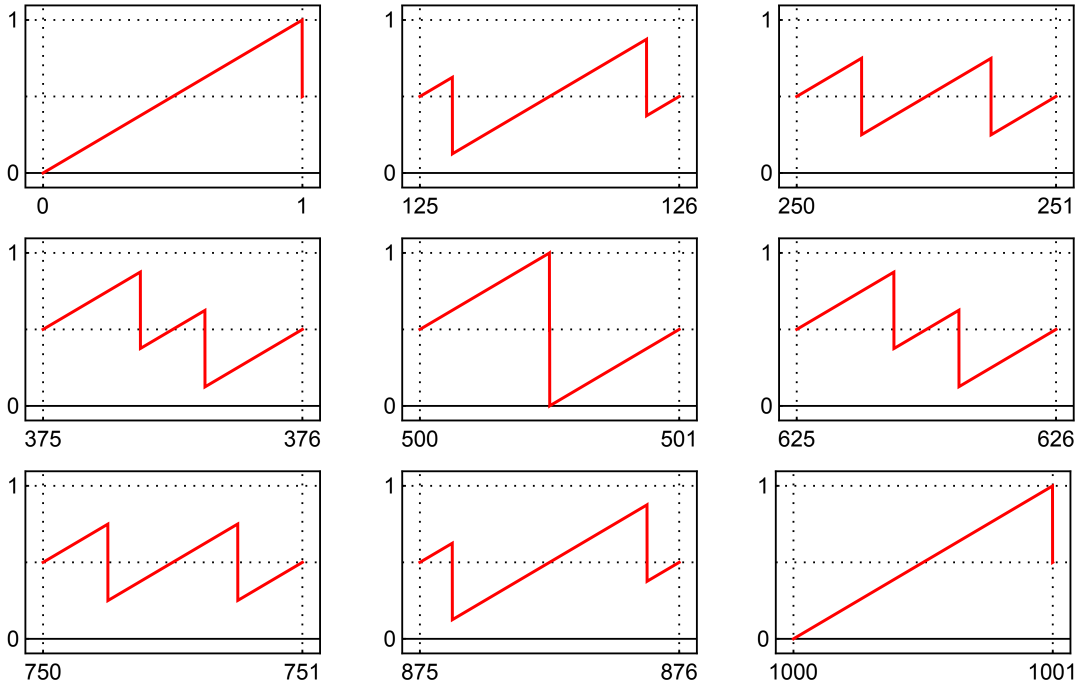

Figure 24: Graphs depicting the wavefroms generated by the function f_1.

Considering the resulting waveforms are piecewise linear, let's try subtracting a sawtooth wave with a unit frequency from the function f_1. This way, the following function is obtained:

$$f_2(x) = f_1(x) - x \bmod 1 \tag{46}$$

As in the previous case, the snapshots of the function f_2 are visualized in Figure 25. Apparently, the subtraction yields waveforms which are piecewise constant. Such waveforms should be trivial to reproduce.

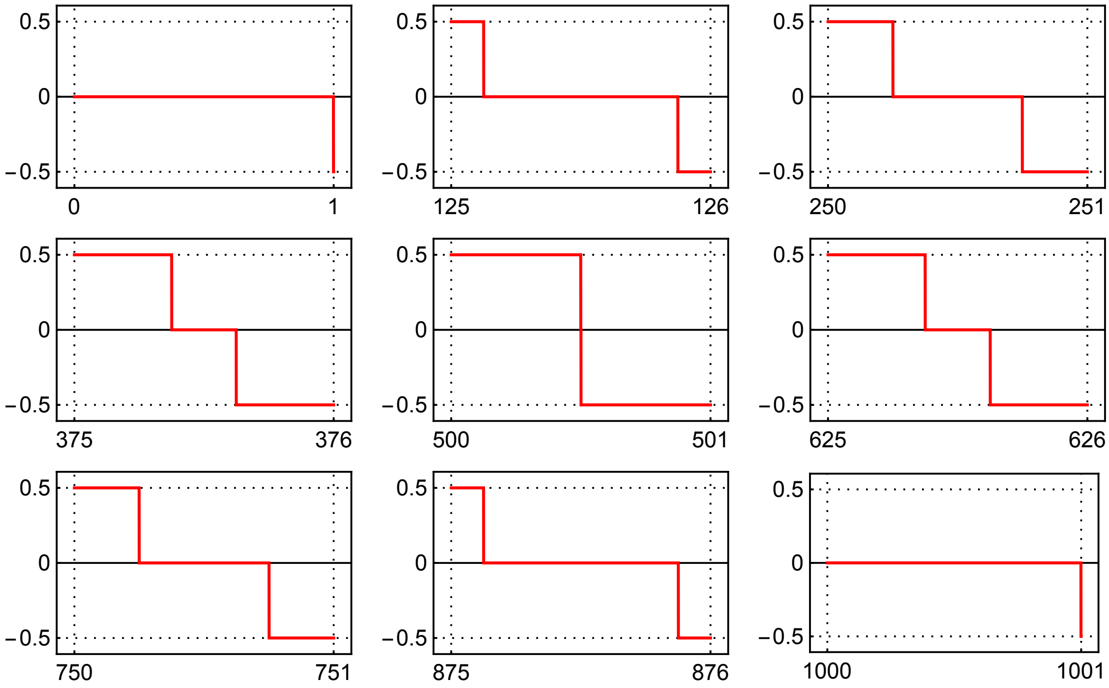

Figure 25: Graphs depicting the wavefroms generated by the function f_2.

In order to reporduce the function f_2, let's consider a difference function g_{diff} defined as follows:

$$g_{diff}(x,m) = \begin{cases} 0.5, & \text{if } x < \frac{m}{2} \\ 0, & \text{if } \frac{m}{2} \leq x < 1 - \frac{m}{2} \\ -0.5, & \text{if } 1 - \frac{m}{2} \leq x \end{cases} \tag{47}$$

where m stands for the modulation parameter. The waveforms generated by the difference function g_{diff} for different values of the modulation parameter m can be observed in Figure 26. Apparently, in comparison with Figure 25, the difference function g_{diff} reproduces only half of the period T_{side}.

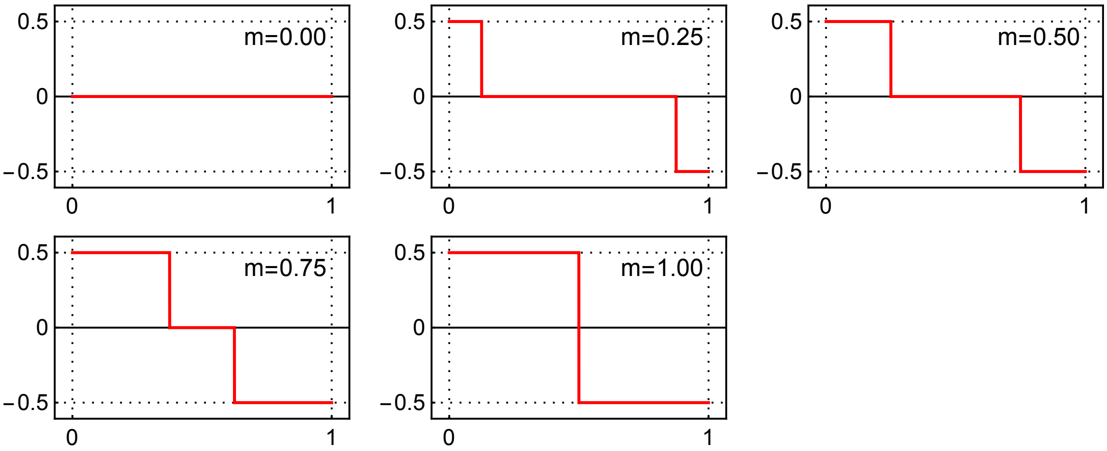

Figure 26: Graphs depicting the wavefroms generated by the difference function g_{diff} for different values of the parameter m.

In order to reproduce the whole period, let's recall the folding function g_{fold} from Equation 15 and define the difference as:

$$y_{diff}(x,m) = g_{diff}\big(x, g_{fold}(m)\big) \tag{48}$$

We can propose reproducing the function f_1 in the following way:

$$f_3(x,m) = x + y_{diff}(x,m) \tag{49}$$

As depicted in Figure 27, it is possible to reproduce the function f_1 using the function f_3, assuming the modulation parameter m is modulated by a sawtooth wave. The frequency of the sawtooth wave should correspond to the period T_{side}.

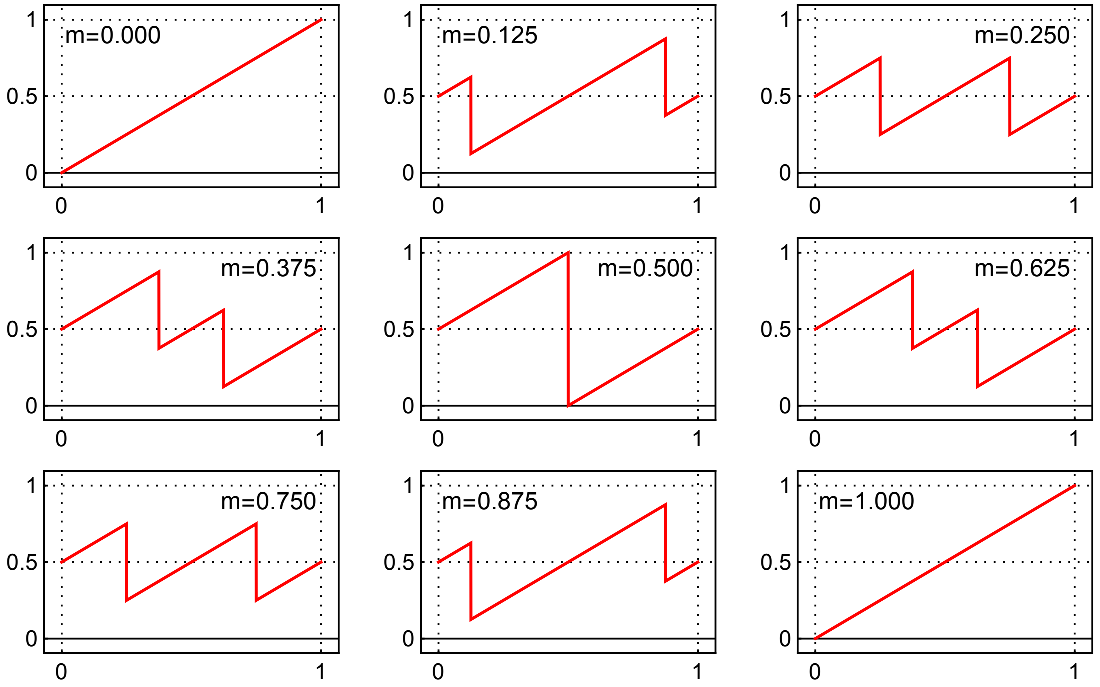

Figure 27: Graphs depicting the wavefroms generated by the function f_3.

For the sake of convenience, let's recall the boundary function g_b from Equation 4 and introduce a sideband function g_{side} defined as follows:

$$g_{side}(x, m, i) = x + y_{diff}\big(x, g_b(i \cdot m)\big) \tag{50}$$

The sideband function g_{side} can be used to generate i-th sideband consisting of two sawtooth waves with the relative frequencies $(1 - i \cdot d)$ and $(1 + i \cdot d)$, each with the amplitude $\frac{1}{2}$. Finally, it is possible to propose a unison function g_{unison} in the following form:

$$g_{unison}(x, m) = \frac{1}{2n + 1} \cdot \big(x + 2 \cdot \sum_{i=1}^{n} g_{side}(x, m, i)\big) \tag{51}$$

The unison function g_{unison} generates $2n + 1$ voices with the relative frequecies $\big(1 - n \cdot d\big), \ldots, 1, \ldots, \big(1 + n \cdot d\big)$, each with the amplitude $\frac{1}{2n+1}$. As proposed, only two phase accumulators are necessary regardless of the number of voices being synthesized, since the unison function g_{unison} has only two inputs.

This approach brings a few advantages. First, even though the unison function g_{unison} should be modulated with a sawtooth wave in order to reproduce the unison effect, it may be modulated in an arbitrary way. For example, it becomes possible to create rhythmic variations in timbre by controlling the modulation parameter m using a sequencer. Also, synchronizing the change in timbre with the song tempo becomes effortless. Lastly, the unison function g_{unison} can be used alongside other phase distorting functions and waveshaping functions.

To illustrate the point, let's consider using the unison function g_{unison} alongside the functions from the previous chapters in the following way:

$$f_4(x, m) = -(g_{sin} \circ g_{dist})\big((g_{rev} \circ g_{unison})(x, m), 1\big) \tag{52}$$

With the number of the detuned sidebands $n = 3$, the waveforms generated by the function f_4 for different values of the parameter m are depicted in Figure 28.

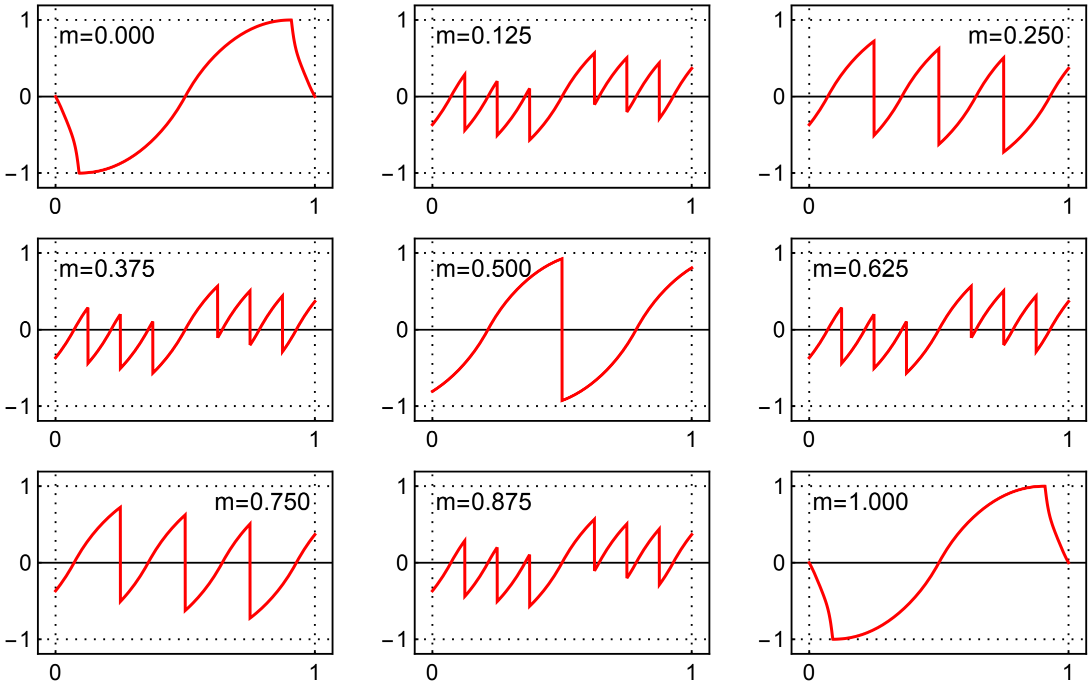

Figure 28: Graphs depicting the wavefroms generated by the function f_4.

7 Conclusion

After introducing the basic terminology, this work explained how to assemble a phase accumulator. A possible way of constructing a sine waveshaping function based upon an iterpolation polynomial approximating the first quadrant of a sine function was shown.

Self-modulation of a sine function was used to create a distorted sine waveform. Also, a phase distorting function capable of reading out a satisfactory approximation of the distorted waveform from the sine waveshaping function was designed.

The effect caused by an interference of slightly detuned sawtooth waves was observed. Subsequently, a method of producing a sawtooth wave unison effect, which requires only two phase accumulators regardless of the number of the detuned sidebands being synthesized, was described. Finally, an example showcasing functions presented in the different chapters being composed together was given.

Future work could be dedicated to obtaining additional distorted waveforms using the process of a self-modulation. Different nonlinear functions could be proposed and the sine function could be substituted. Alternatively, an emulation of the Supersaw oscillator found in Roland JP-8000 could be built based on the sawtooth wave unison effect presented.

References

[1] KLEIMOLA, Jari, Victor LAZZARINI, Joseph TIMONEY and Vesa VÄLIMÄKI. Phaseshaping Oscillator Algorithms for Musical Sound Synthesis. *SMC Proceedings of the 7th Sound and Music Computing Conference, Barcelona, Spain, July 21-24, 2010.* 2010. Available at: http://smcnetwork.org/files/proceedings/2010/15.pdf

[2] SZABO, Adam. *How to Emulate the Super Saw.* Stockholm, 2010. Bachelor's thesis. Royal Institute of Technology, School of Computer Science and Communication.