EPRO/IAS

Bachelor Studiengang Medientechnik
Fachhochschule St.Pölten

Patrik Lechner

Wien, January 3, 2017

1. Introduction

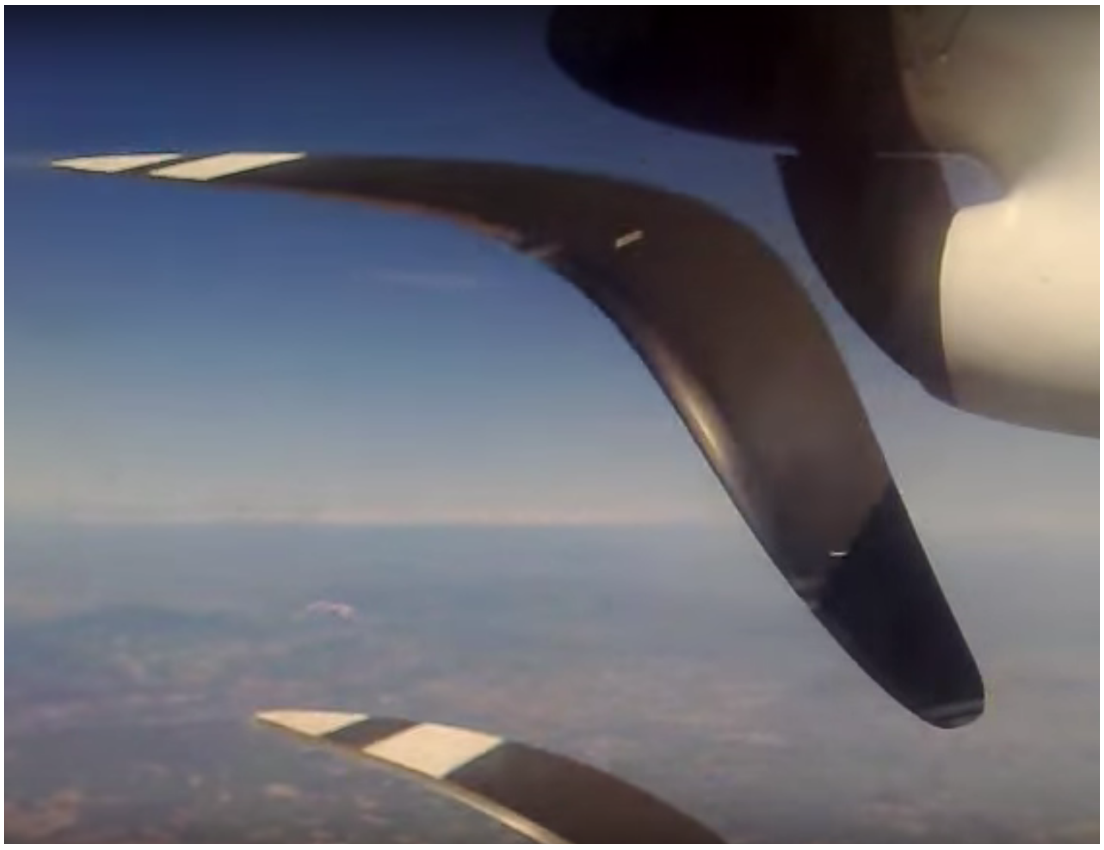


Figure 1.1.: Weird effects of digital signals in video

This section tries to make sure we are all on the same page. It goes through some mathematics notation and principles of digital audio.
If you rather want a video format revision of digital audio (I mean, it's 2017, of course you want), I can recommend D/A and A/D | Digital Show and Tell (Monty Montgomery @ xiph.org) [1]. Some of the points in the video go beyond what we are doing here and vice versa, so don't purely rely on the video though.
Ideally, if you throughly understood the introduction section, you should be able to go through the remaining chapters with tempo.

1.1. About this document

Some information in this document is relevant for understanding it's contents but not relevant for the exam. For example in chapter 2 we will find a really complicated result of an equation. The point there is just that it's complicated. And it is shown how complicated, but this is nothing to be learned by heart.
Please report any mistakes, errors etc to ptrk.lechner@gmail.com.

Text and equations with a gray background like this are background information that is not to be learned by heart.

Video analogies
This document tries to explain digital signals. It does this by use of audio signal mainly. Sometimes video analogies are given. These are also not relevant for the exam.

1.2. About plotting signals

We will need to plot a lot of signals in order to understand them better. Most of the time, such a plot will look like figure 1.2.
This plot looks nice but it has a problem. The sine wave is sampled at a sampling rate of 30 Hz, but we see a continuous line. This "connection of the dots" is created by plotting. It is kind of similar to what our *digital-to-analog-converter* does. It somehow[2] interpolates the values we have.
This can be misleading, so we should actually plot something more like figure 1.3. Often we can see plots that look like figure 1.4 as well when a signal is analyzed.

[1]https://www.youtube.com/watch?v=cIQ9IXSUzuM
[2]linear interpolation in case of the plot

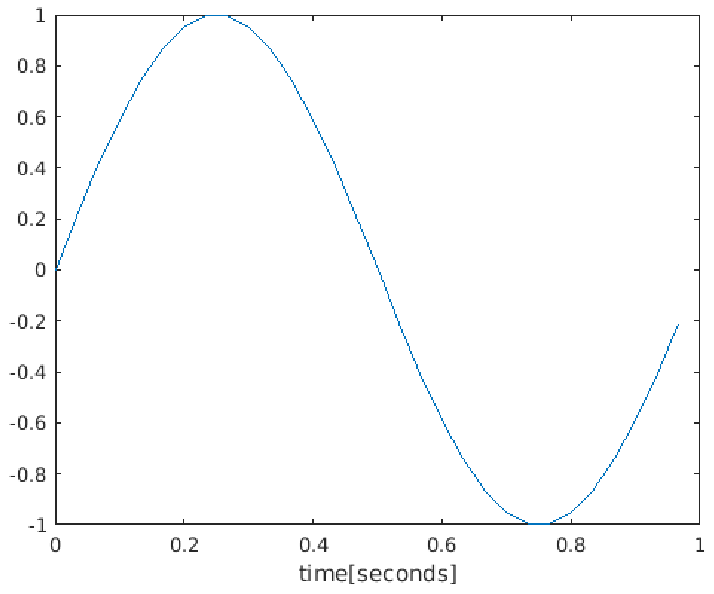


Figure 1.2.: Sine wave, 1Hz, sampled at 30Hz sample rate

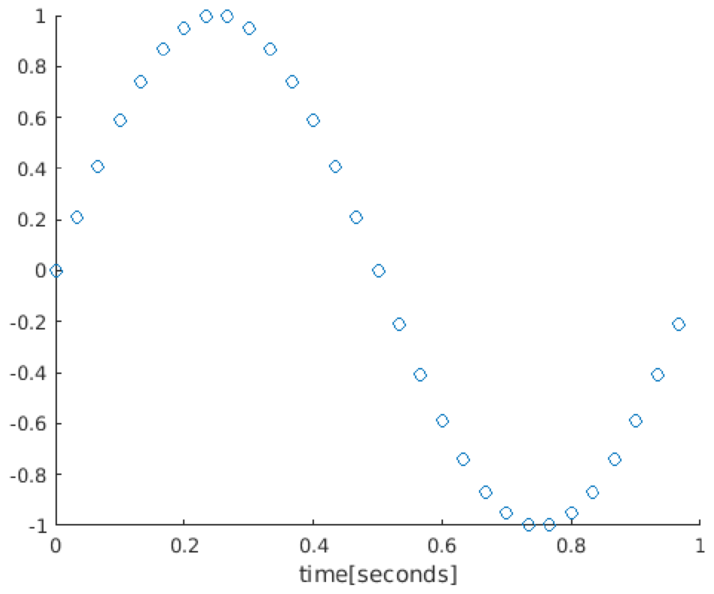


Figure 1.3.: Sine wave, 1Hz, sample rate 30Hz, scatter-plot

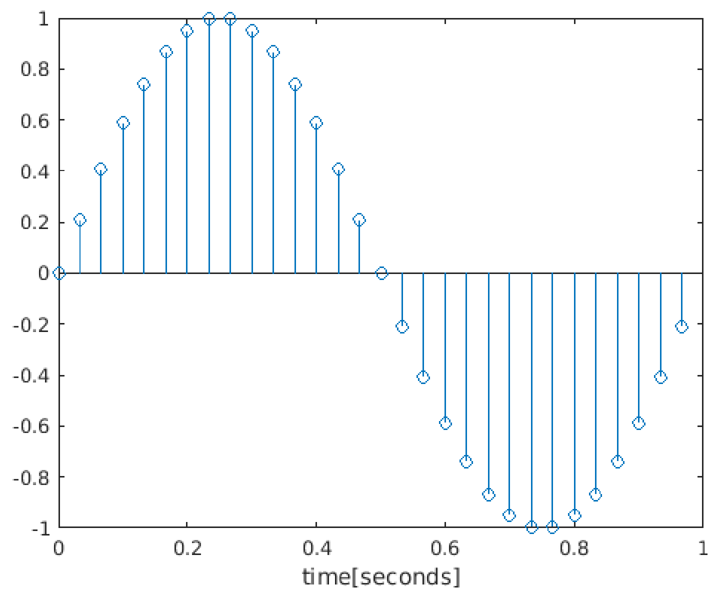


Figure 1.4.: Sine wave, 1Hz, sample rate 30Hz, stem-plot

So why don't we always do a stem- or scatter-plot? Simply because it gets too crowded with our usual sampling rates in audio. It just works with very low sampling rates or very short signals.

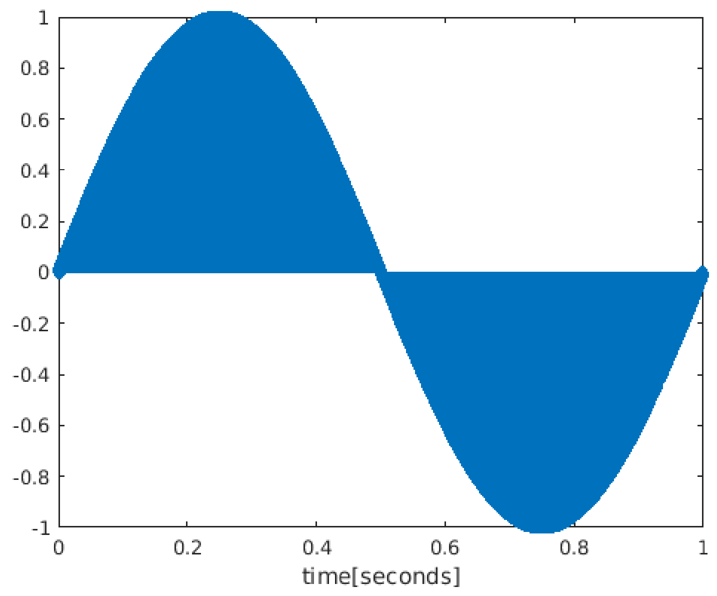


Figure 1.5.: Sine wave, 1Hz, sample rate 44100Hz, stem-plot

But we should never forget that we don't actually have the values in between

the dots. Digital signals are not defined between their sampled points.

1.3. What is aliasing?

Aliasing in audio means problems caused by signals that exceed the nyquist rate.
The nyquist rate,let's call it f_n for now, is defined by the half of the sample rate (f_s).
So,

$$f_n = \frac{f_s}{2} \tag{1.1}$$

A digital system can only describe signals up to his nyquist rate. If we try to make signals higher than this frequency, we will fail and encounter strange effects.
Visually speaking, frequencies higher than nyquist fold back. So, let's assume we have a sampling rate of 100Hz. Nyquist would be at 50Hz. If we try to synthesize a sine wave with 51Hz, what we will get is a 49Hz one. If we try to make a 52Hz one, we will get 48Hz. So you see, it simply folds back.

Video analogies
Aliasing in graphics usually means *spacial* aliasing, so aliasing in the space domain. This is what we see in figure 1.6. But there is also time domain aliasing in film. It is actually more natural to think about the sampling rate in audio as the same as the frame rate in video. For some really weird effects that arise in video due to time domain aliasing see airplane[3], Water experiment[4]or Guitar strings[5].

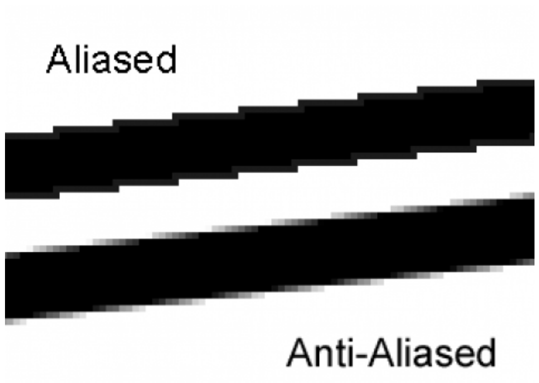


Figure 1.6.: Spacial aliasing in graphics. The "frequency" of the intended pixels is to high for the actual pixels.

The particularly strange effects in the airplane example above are caused by the rolling shutter of a CMOS sensor. Since it is not sampling the incoming light uniformly (at the same time) the image is distorted.

In fugure 1.7 you can see a visualization of aliasing in the time domain.

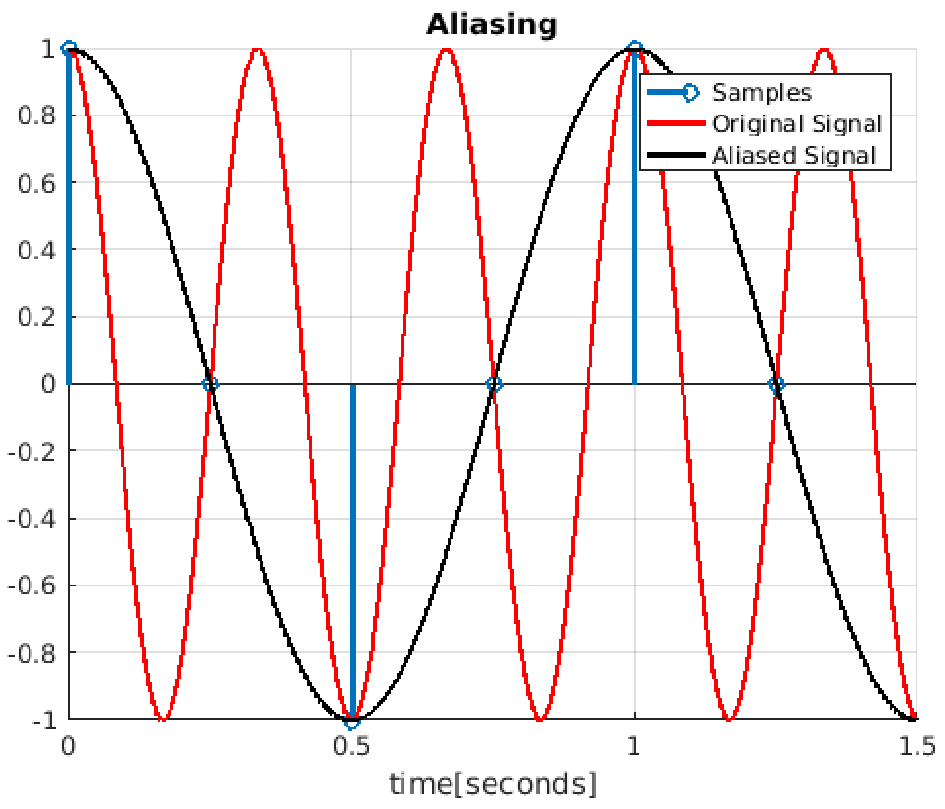


Figure 1.7.: Aliasing visualized in the time domain. A sampling rate of 4 Hz is used. Therefore frequencies will fold above 2 Hz, which is the nyquist frequency. The input cosine has a frequency of 3 Hz, labeled “Orginal Signal”. What we would get is the sampled points. If these are digital-to-analog converted, we would get what is labeled “Aliased signal ” in this plot, so a 1 Hz cosine.

1.4. Scaling and Mapping Signals

It is an important skill to be able to scale signals from one range to another. We need it a lot and we will be able to think about signals more easily if we mastered this task. It’s actually quite simple, we just have to imagine the signals visually.
So what exactly do we have to do here? We are confronted with the following problem: Given some signal, say, a sine wave with its maximum at the value 1 and its minimum at the value -1. How to bring it to a different range, say, 0-10?
It helps me a lot to solve this problem in two parts: first get the input in the range 0-1,

[2]https://www.youtube.com/watch?v=LVwmtwZLG88
[3]https://www.youtube.com/watch?v=GBtHeR-hY9Y
[4]https://www.youtube.com/watch?v=jcOKTTnOIV8

then from there go to the desired range. What can we do to the signal? Let's take a sine wave:

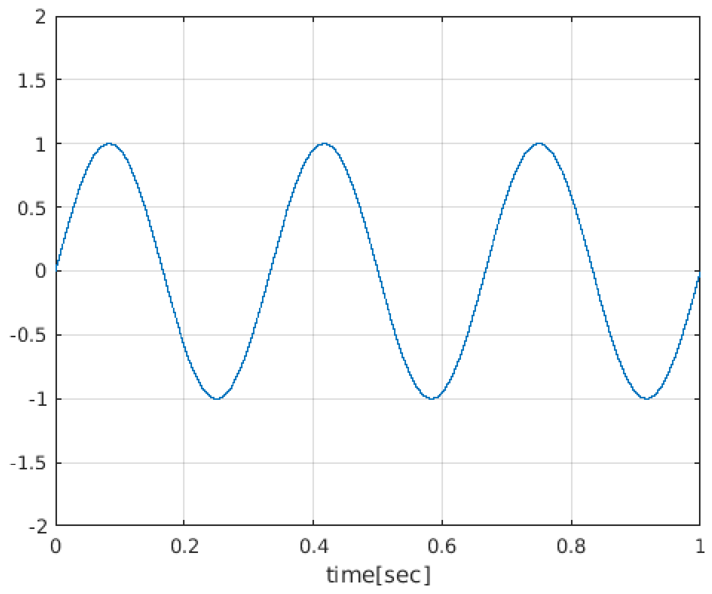


Figure 1.8.: a sine wave

Well we can add and subtract to move the wave vertically, so let's add 1 to move it up:

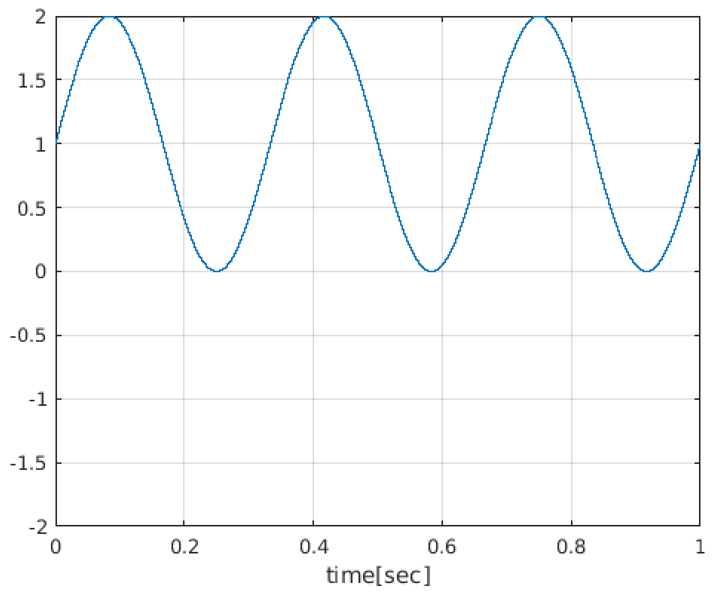


Figure 1.9.: the same sine wave, 1 added to a each sample, therefore shifted upwards.

So we can move signals around by adding constant values. We can scale them by multiplication. So if we take our sine that now ranges from 0 to 2 and multiply it by 0.5, we get whats in figure 1.10.

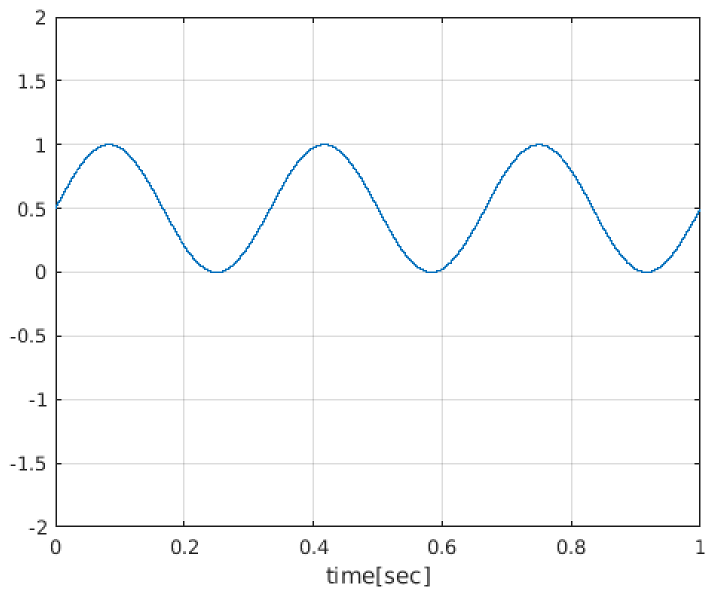


Figure 1.10.: Sine with a range of 0-1. Obtained by taking a sine wave, adding 1 and dividing by two afterwards.

Using what we got now in figure 1.10, we can just multiply by 10, easy! Beware that there are always multiple solutions to this kind of a problem. Try to find another one for the problem above!

Question 1 *Let's take a sine wave that has it's minimum at 2 and it's maximum at 5. What do we have to do to get it into a -1 to 1 range?*

Answer 1 *We could subtract 1.5 to center the wave around zero first. Afterwards we take care of the amplitude by multiplying by $\frac{2}{3}$ (since the initial wave has a peak-to-peak amplitude of three and we want a peak-to-peak amplitude of 2)*

1.5. What's DC-Offset?

What we did above by adding a constant value to a signal can be called adding DC-offset ("Gleichspannungsversatz"), DC-Bias or a DC component. These are different words for the same thing.
DC-offset can also be encountered in signals we recorded (caused by old or broken equipment mainly). But we have seen that we can also generate DC-offset.
work in progress.

1.6. What's an Impulse?

work in progress.

1.7. How to describe audio mathematically

If we want to talk about signals, or if we want to analyze them, it is often useful to look at the problem mathematically. First, let's introduce some conventions. They might look unfamiliar or complicated. But in fact it is not very complicated and knowing these conventions makes it easy to communicate (e.g. reading scientific papers about our topics or explaining something to another person).

1.7.1. signals

Usually we describe a *digital* signal by a name, say x, (but you can call it however you want). If we want to talk about the individual samples, or values of the (mono) signal, we can use a subscript or parenthesis. So if the fith sample of x is 1, we could write:

$$x_5 = 1 \tag{1.2}$$

or

$$x(5) = 1 \tag{1.3}$$

Oftentimes we like to talk about a signal more generally and we use n as a place holder for this index, so we might write x_n, meaning the nth sample of x.

sine waves

We use sine waves a lot. The syntax can get a bit overwhelming at first, so let's quickly explain what's going on in a standard sine wave oscillator.

$$x(t) = A \cdot sin(2\pi ft + \phi) \tag{1.4}$$

Where f is the frequency in Hertz and t is the time in seconds. ϕ Is a (possibly constant) phase offset and A can be used to scale the whole thing. Since cosine and sine have their peaks at -1 and 1, A will be the amplitude. If A is set to 0.5, the resulting signal will have it's peaks at -0.5 and 0.5. work in progress.

1.7.2. systems

1.8. Message Domain/Signal Domain

work in progress.

Contents

Part I.

Semester 3

1. Sampling, waveshaping, and non-linearity

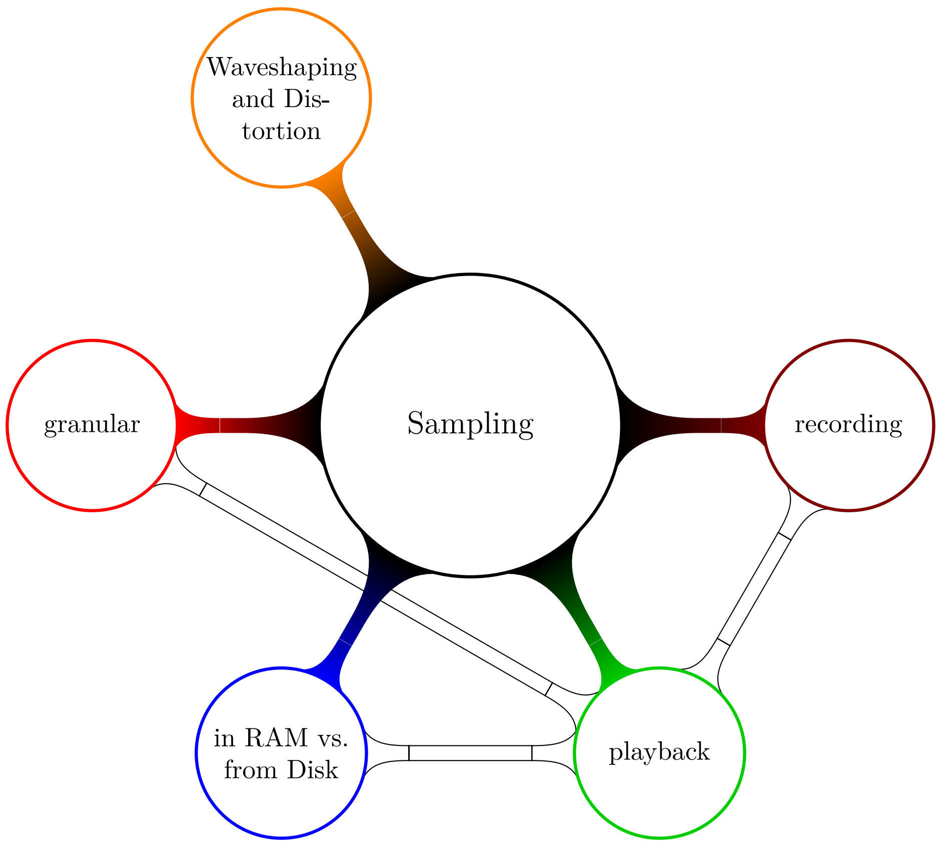


Figure 1.1.: Lecture Contents

1.1. Waveshaping

Wikipedia quote, page “wavesahper”:
„The mathematics of non-linear operations on audio signals is difficult, and not well understood.“ Waveshaping means distortion. It adds overtones, take a look at figure 1.2.

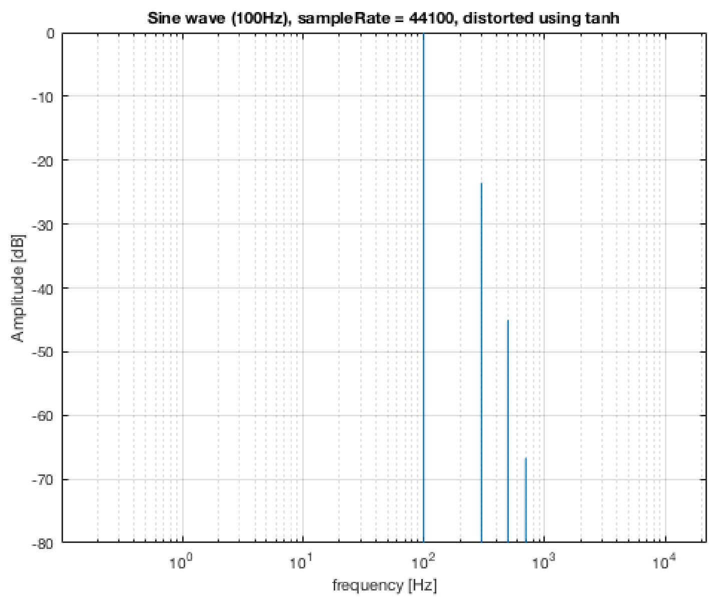


Figure 1.2.: A sine wave has been generated and waveshaping was applied to add overtones.

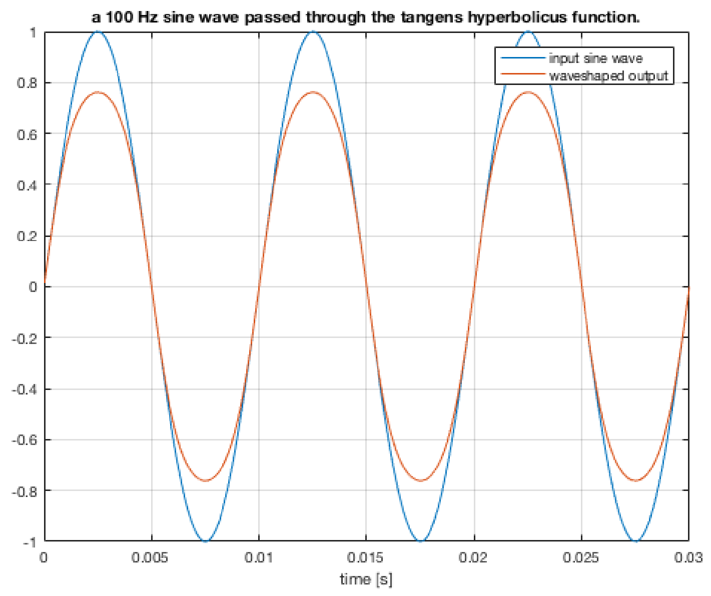


Figure 1.3.: The same as the spectrogram above, but in the time domain. We can see the input sine wave and the slightly distorted output. It may look like just the amplitude has changed, but the sine's actual *shape* has changed slightly

1.1.1. The simplest case: a linear Transfer function.

See 1.4. A linear transfer function is used as a lookup table for a sinosoidal input.

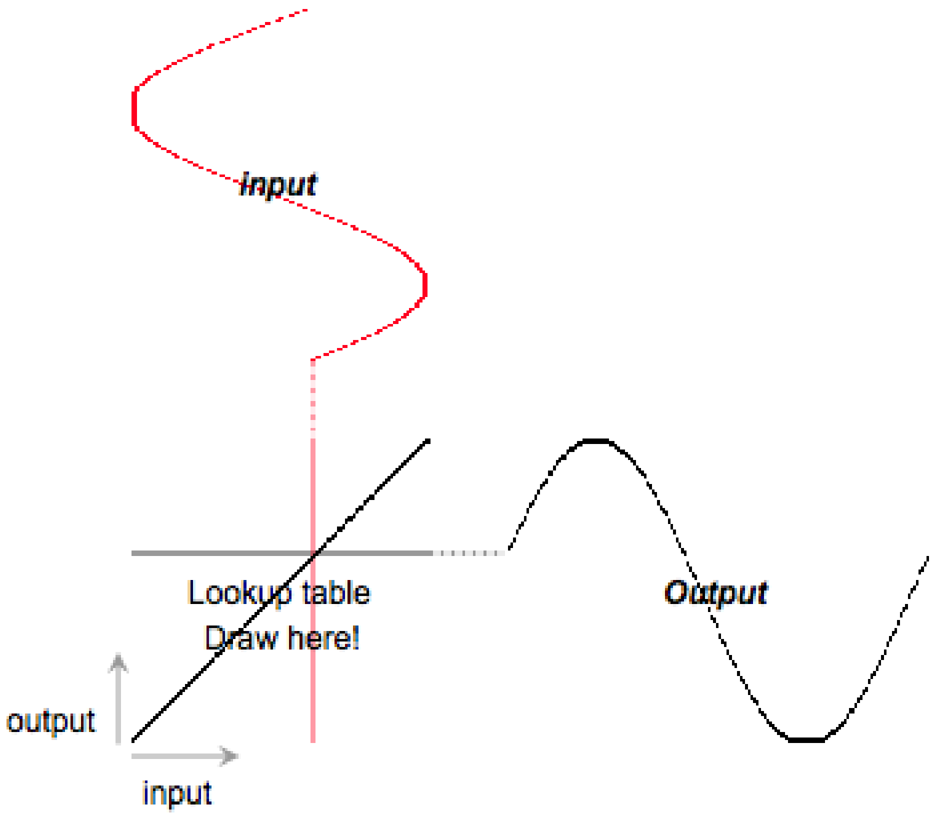


Figure 1.4.: Linear Transfer function

A transfer function in the sense of a waveshaper (a “transfer function” might also mean frequency response in other contexts) is a simple look-up function. Waveshaping means to use an input wave to *look up* values in a table or function. A linear transfer function, let’s call it l, can result in no change, for example, it might return $l(x) = x$. This means, that whatever value we pass in, we get the same value out. Other linear transfer functions might *only* change the amplitude. For example $f(x) = x \cdot \frac{1}{2}$. That doesn’t seem very interesting. But it might explain the term “linear”. A transfer function is linear if it looks like a line if we plot it. Look at figure 1.5.
Non-liner transfer functions behave differently. They map their input to other values, such as $f(x) = x^2$. And if we plot then, they don’t look like a line. You can also look at figure 1.6 in order to understand what’s happening. We again see a linear transfer function but also a non linear one.

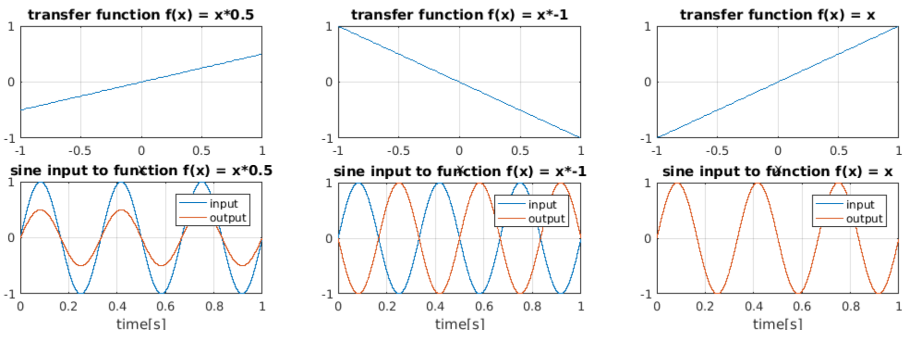


Figure 1.5.: A couple of linear transfer functions and their corresponding effects demonstrated using a sine wave. From left to right: multiplication by 0.5, so attenuation by about 6dB, inversion, and the "do-nothing"-function.

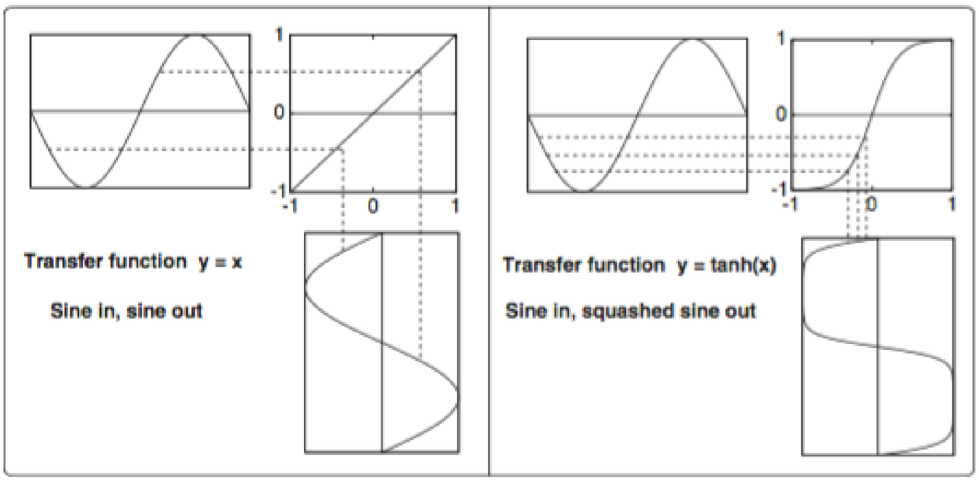


Figure 1.6.: A waveshaping visualization taken from Farnell (2010)

But let's get back to our square function, since it's simpler and we will find some surprising results when analyzing it. Let's first simply plot it too.

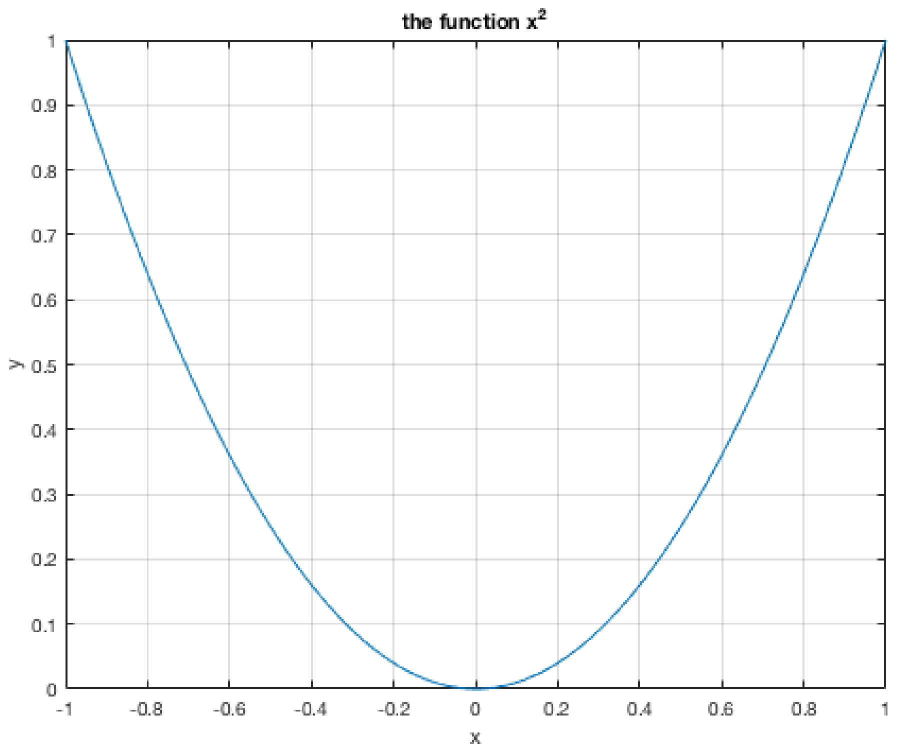


Figure 1.7.: The function $f(x) = x^2$

Video analogies
What does applying a curve to every pixel of a video or image do? Is this even something people do? Of course! Think of contrast curves, below is an example from photoshop. Beware that video is working with unipolar input values and audio with bipolar inputs.

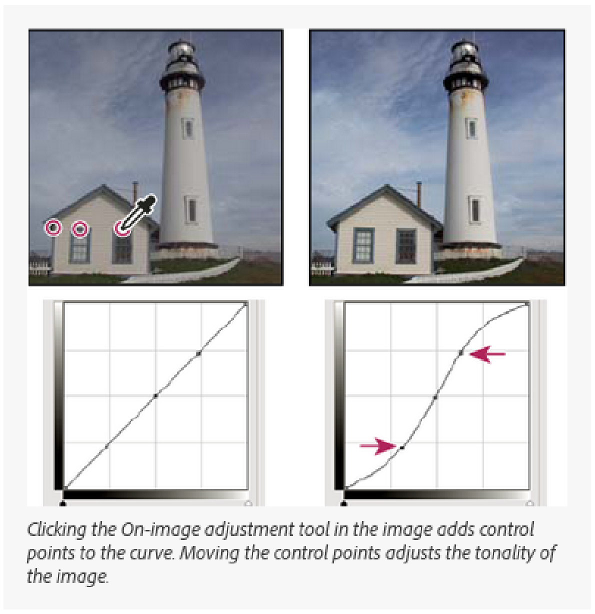

Clicking the On-image adjustment tool in the image adds control points to the curve. Moving the control points adjusts the tonality of the image.

Figure 1.8.: Contrast curve in photoshop

1.1.2. Simple non-linearity: X^2

So let's analyze what happens if we use this function for waveshaping. Here it is again:

$$f(x) = x^2 \tag{1.1}$$

Let's simply listen to what's happening, building it in pd:

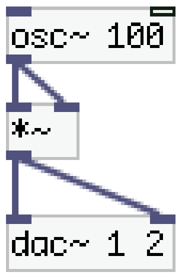


Figure 1.9.: The square function in pure data, using a 100Hz sine as a test signal. What do you hear?

And we can simply plot what happened if we apply the function before also trying to understand analytically:

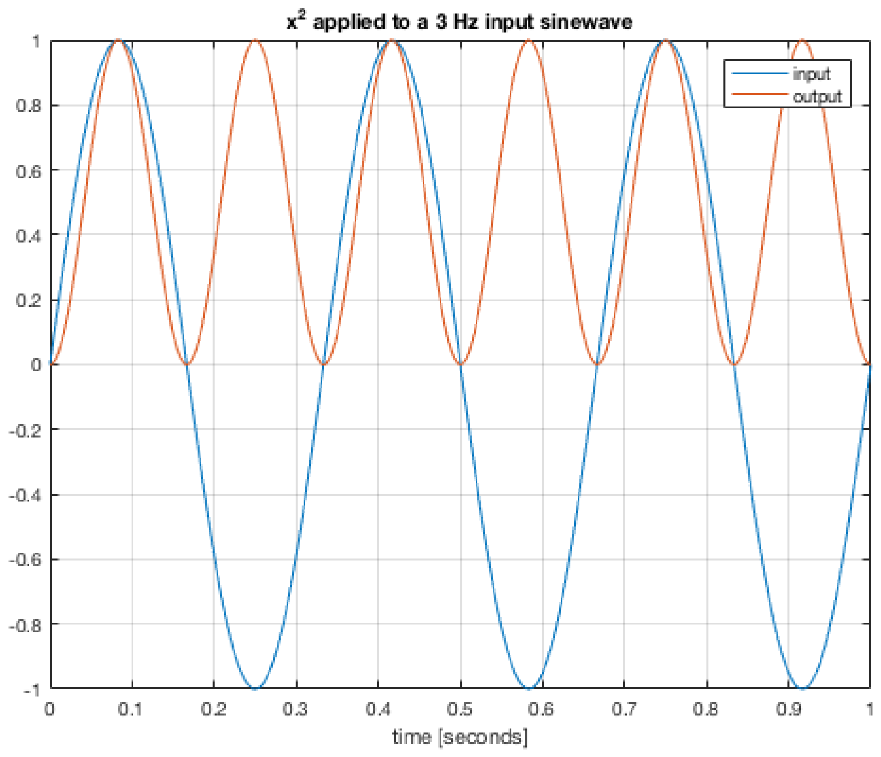


Figure 1.10.: Applying the square function to an input sine wave.

Weird, the input seems to double in frequency (did you hear that?). Let's try to understand what's happening.
So we calculate what happens if we send a cosine through this function, so let's take:

$$x = cos(\omega) \tag{1.2}$$

wit arbitrary ω. We can just ignore ω here for a while. Usually, there should be some indexing variable in the cosine function if we want to describe an oscillator that moves over time, but let's also skip that.
So applying our square function we of course get:

$$y = cos(\omega)^2 \tag{1.3}$$

This again results in:

$$y = cos(\omega) \cdot cos(\omega) \tag{1.4}$$

So far so trivial. Note that a multiplication of two oscillators is called *Amplitude Modulation* (actually, in this case we encounter "Ring Modulation", but let's ignore that also), and we know things about Amplitude modulation, namely:

When multiplying two oscillators, we get sum and difference of the two input frequencies. (And the whole output is attenuated by 6dB)

The above statement in equation form:

$$cos(a) \cdot cos(b) = \frac{cos(a+b) + cos(a-b)}{2} \tag{1.5}$$

We could also have looked up this *trigonometric identity*. This means for our experiment with our cosine squared:

$$y = \frac{cos(\omega + \omega) + cos(\omega - \omega)}{2} \tag{1.6}$$

So:

$$y = \frac{cos(2 \cdot \omega) + cos(0)}{2} = \frac{cos(2 \cdot \omega)}{2} + \frac{1}{2} \tag{1.7}$$

We arrive at the same result! But is this true for every input? That would mean we just built a frequency shifter, did we? No. Waveshaping is much more complicated.

This is immediately obvious when we try to do the same with two oscillators:

$$x = cos(\omega_1) + cos(\omega_2) \tag{1.8}$$

then

$$y = (cos(\omega_1) + cos(\omega_2))^2 \tag{1.9}$$

$$y = cos(\omega_1)^2 + cos(\omega_2)^2 + 2 \cdot cos(\omega_1) \cdot cos(\omega_2) \tag{1.10}$$

And finally:

$$y = \frac{cos(2 \cdot \omega_1)}{2} + \frac{1}{2} + \frac{cos(2 \cdot \omega_2)}{2} + \frac{1}{2} + 2 \cdot (\frac{cos(\omega_1 + \omega_2) + cos(\omega_1 - \omega_2)}{2}) \tag{1.11}$$

1.1.3. How can waveshaping be implemented?

Take a look at figure 1.11. What do you think is happening? On the left side, we see waveshaping as we did it above, using a mathematical function, in this case the tangens hyperbolicus, to distort our signal. On the right side, we see a table that contains the authors desperate attempt to draw the same function with the mouse. The results are theoretically equivalent (if the function in the table was correct), but what are the advantages and disadvantages of the two approaches? Also, be sure to understand what the addition of 1 and the multiplication with 50 does on the right side. Hint: the array has 100 points.

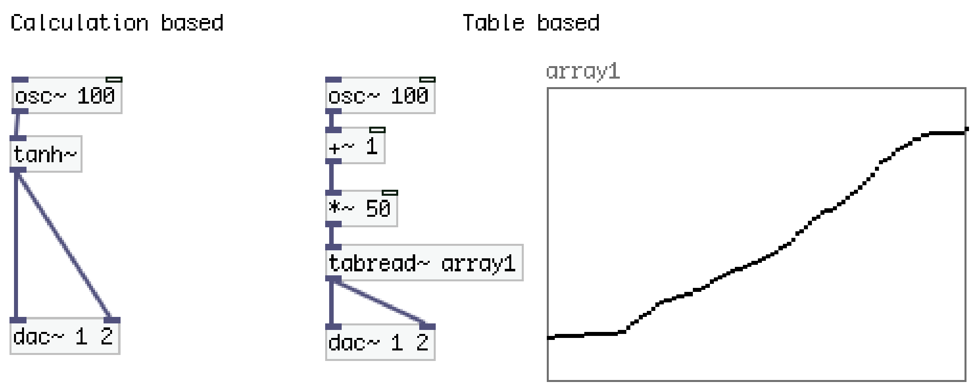


Figure 1.11.: Left: using a mathematical function to calculate the output. Right: using a table to look up the output.

1.1.4. How is Waveshaping related to other techniques?

Sampling

If we take a look at figure 1.19, we see that we play a sound file by accessing a buffer (wavetable) using an index, an oscillator. This is effectively the same setup as we would build for distorting an input sound. Also take a look at figure 1.12, which showing us that waveshaping and wavetable synthesis are identical.

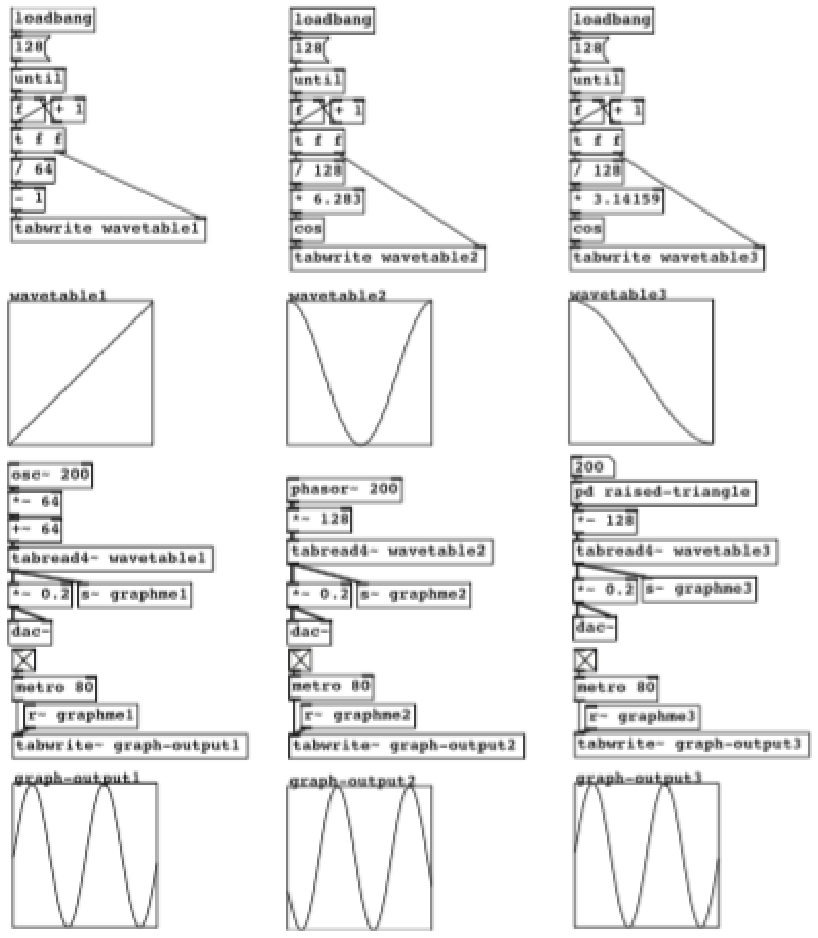


Figure 1.12.: Picture taken from Farnell (2010), showing the identity of waveshaping and wavetable synthesis

Modulation

While we will talk about modulation in a separate chapter, let's loosely define amplitude modulation (AM) as the multiplication of two oscillators and frequency modulation (FM) as varying the frequency of an oscillator using another oscillator. So, as we have also seen above, AM looks like this:

$$y = sin(a) \cdot sin(b) \tag{1.12}$$

and FM looks like this:

$$y = sin(sin(a)) \tag{1.13}$$

in practice, the a and b terms are a bit more complicated, but we will look at this later. That certain cases of AM are identical to waveshaping has been shown above, think about the square function again. This of course does not mean that waveshaping can do everything AM can do and it does not mean that AM can achieve everything that waveshaping can. This should just show that we can understand the techniques from the perspective of another. What about FM? Well if our lookup function we use for waveshaping is a sine wave, we arrive at the exact same equation as how we defined FM above. Again, practically speaking, the results we get with these two techniques are very different, but we can see the connections.

1.1.5. Why is Waveshaping useful?

The output spectrum is dependent on the input amplitude. This makes it easy to create complex evolving spectra.

1.1.6. What are the problems with waveshaping?

Waveshaping adds overtones. When we build a waveshaper, we have to be aware of aliasing. Take a look at figures 1.13 and 1.14. Sinewaves have been amplified and clipped here.

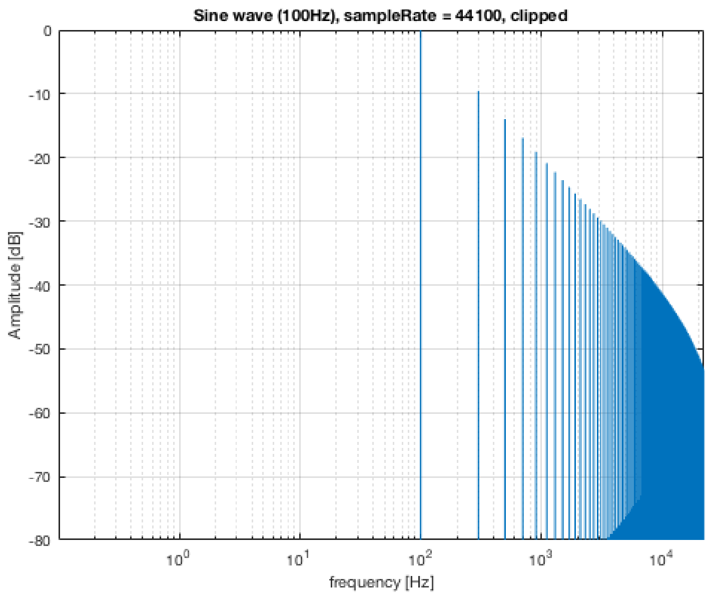


Figure 1.13.: A sine wave was generated and clipped. Clipping is a form of waveshaping which adds many overtones. Note how high frequencies fold back into the lower parts of the spectrum because they exceed the Nyquist-rate.

In pd we could achieve this like in figure 1.15.

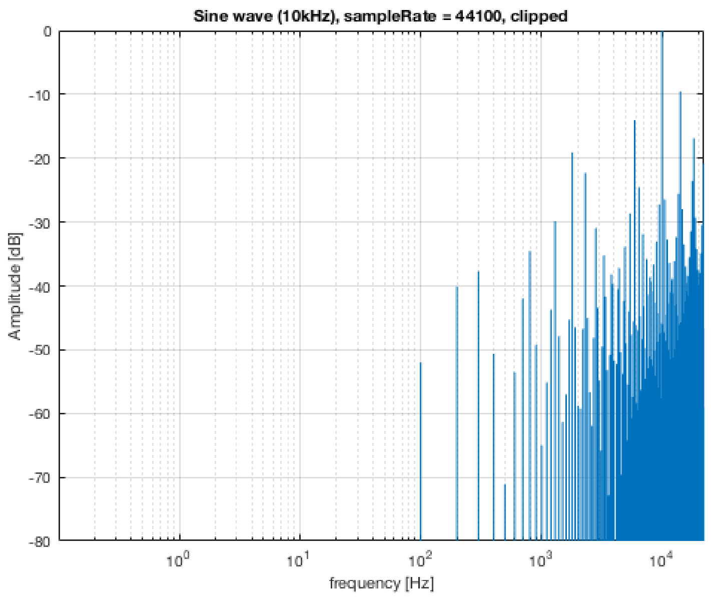
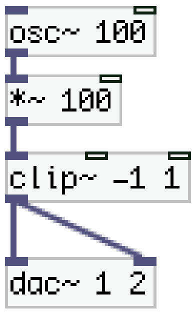


Figure 1.14.: Again, a sine wave, this time with a higher frequency to begin with. Extreme clipping has been applied by boosting the input amplitude. The aliased overtones are all over the place, even below the input frequency.



```
osc~ 100
*~ 100
clip~ -1 1
dac~ 1 2
```

Figure 1.15.: Clipping an amplified sine wave in pd

What does the output look like? Let's not only look at the spectra but also at the time signal:

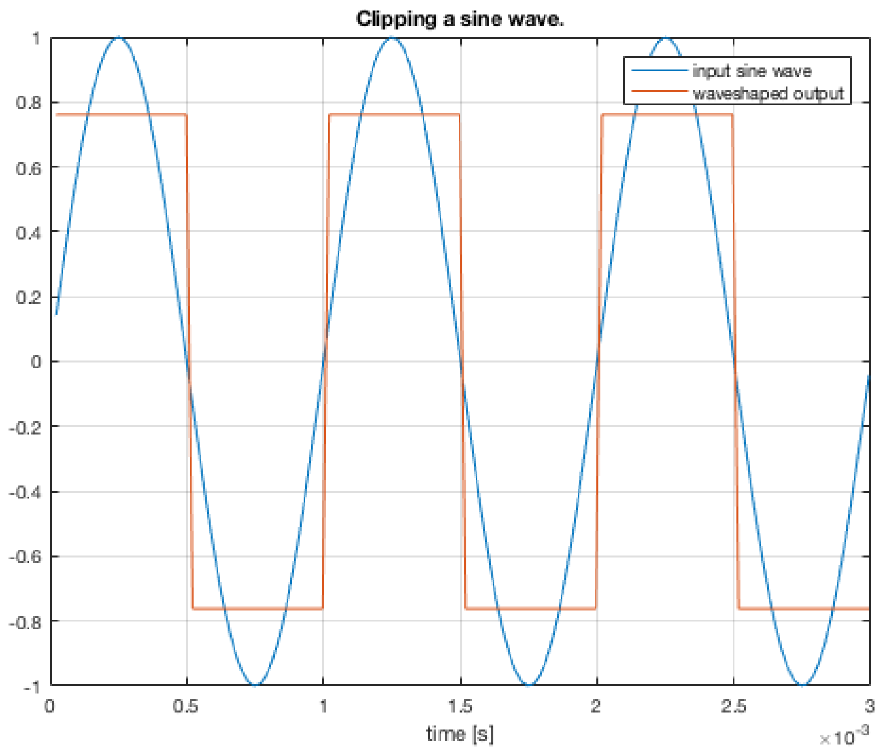


Figure 1.16.: An amplified and clipped sine wave in the time domain.

We see that we can arrive at a square-wave like result, but this square-wave is not anti-aliased.

The problem of aliasing in waveshaping is usually treated by over-sampling. This does not solve the problem but lessens it significantly resulting in cleaner, arguably better sound. Oversampling means that, if we work at a sample-rate of 44.1kHz, the input is up-sampled, essentially interpolated, to be at a sample-rate of 88.2kHz. Then the waveshaping is applied, leaving room for high frequencies up to 44.1kHz. Using a lowpass filter, high frequencies over 22.1kHz are then attenuated as much as possible, in order to be able to down-sample again to reach our initial sample-rate of 44.1kHz. To state it more simple: Waveshaping is usually encapsulated in a process that runs at higher sampling rates in order to lessen aliasing.

1.2. Sampler

Work in progress.

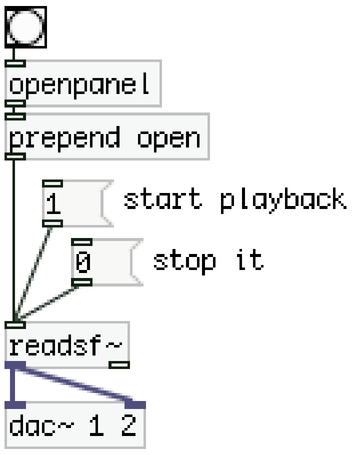


Figure 1.17.: simpleSampler

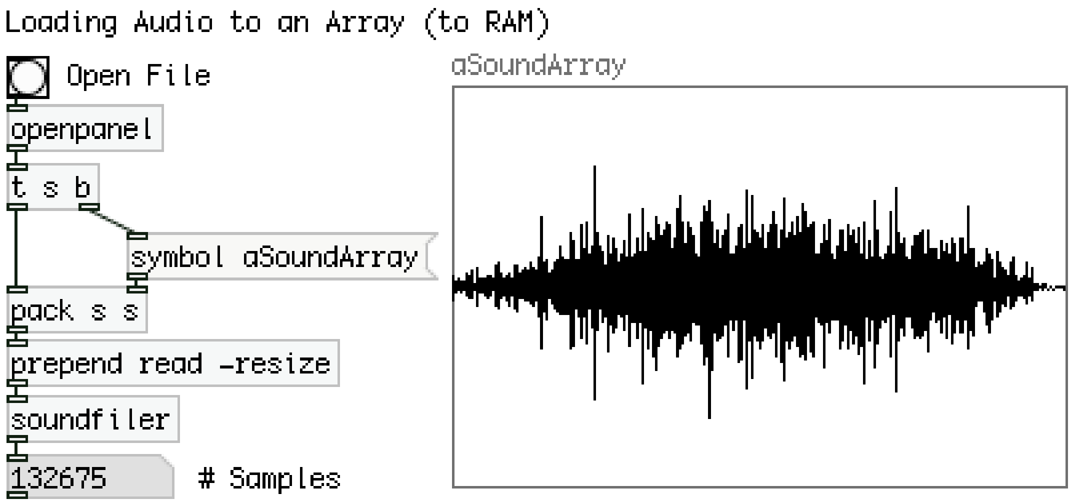
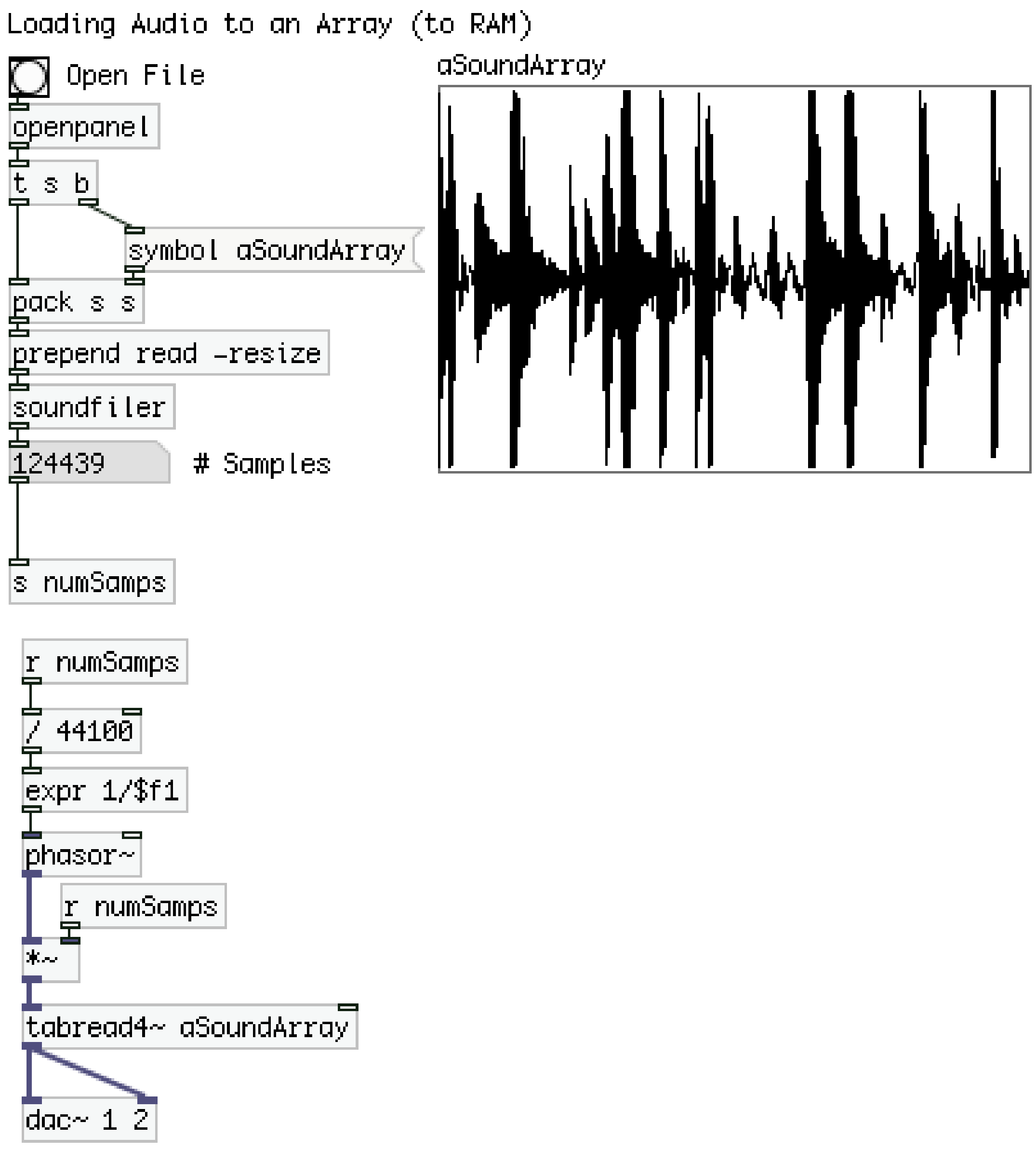


Figure 1.18.: sound in Ram



```
Loading Audio to an Array (to RAM)
Open File
openpanel
t s b
symbol aSoundArray
pack s s
prepend read -resize
soundfiler
124439  # Samples
s numSamps

r numSamps
/ 44100
expr 1/$f1
phasor~
r numSamps
*~
tabread4~ aSoundArray
dac~ 1 2
```

Figure 1.19.: RamFilePlayback

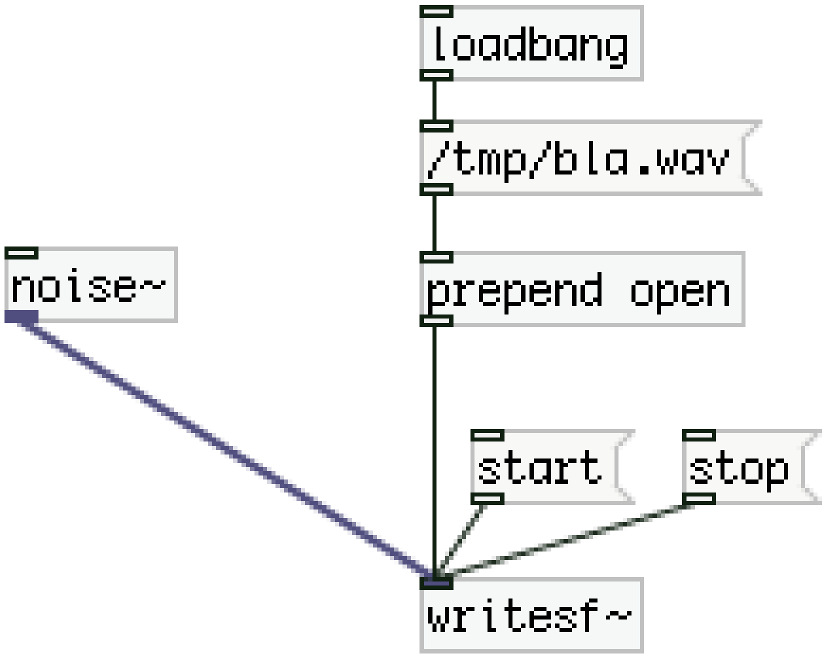
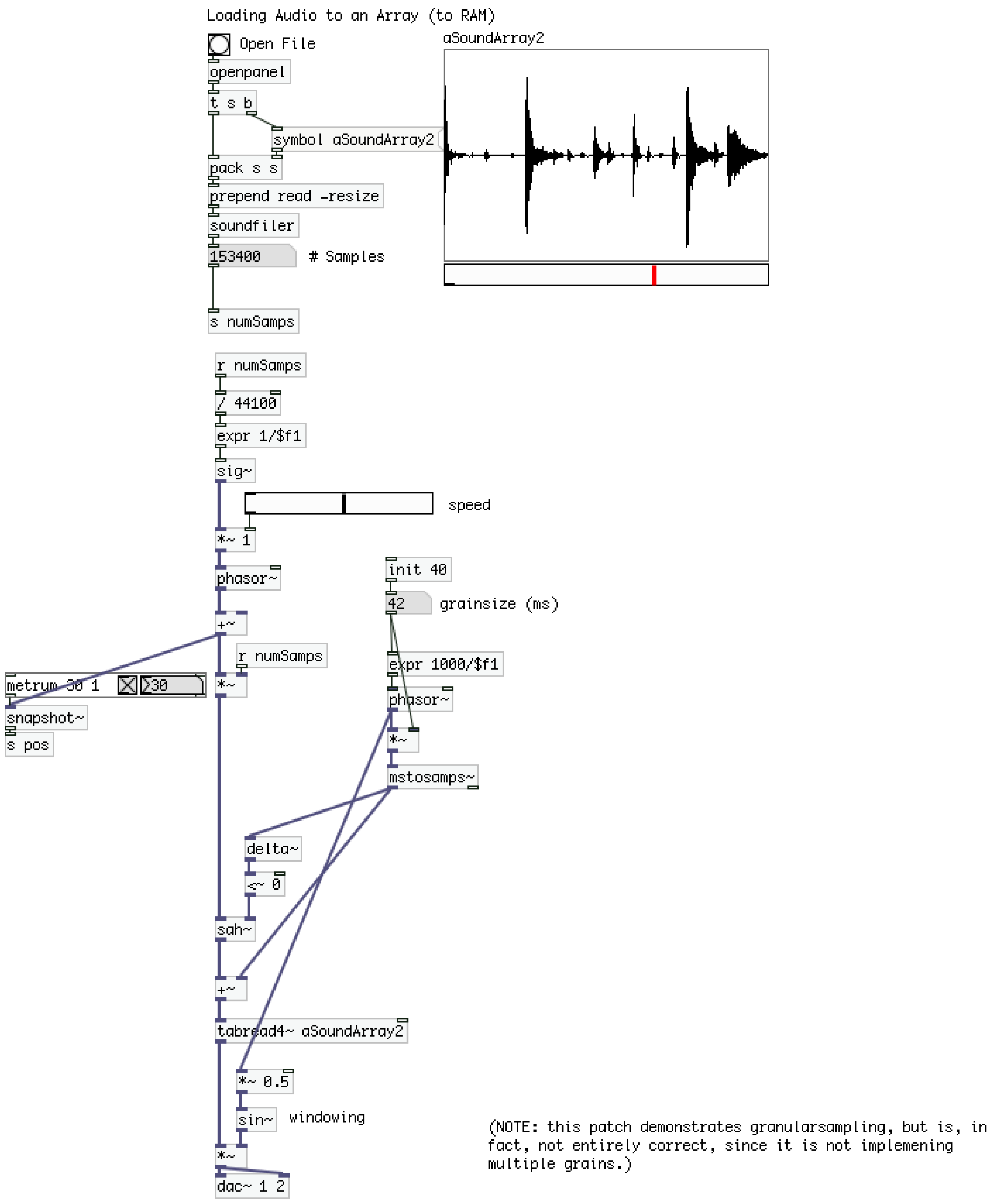


Figure 1.20.: writing Audio to disk



(NOTE: this patch demonstrates granularsampling, but is, in fact, not entirely correct, since it is not implemening multiple grains.)

Figure 1.21.: moreSamplimg.pd, a simplified version of granular sampling

1.3. Hausübung

1.3.1. Testmodul

baue ein audio Testmodul mit folgener spezifikation:

- Ein audio output
- verschiedene klangquellen wählbar:
    1. White Noise
    2. Sinus (freq. einstellbar)
    3. soundfile (file wählbar)
- GUI
- verfügbar(in eurem pfad, und jederzeit abrufbar als abstraction)
- output pegel sichtbar (level meter)

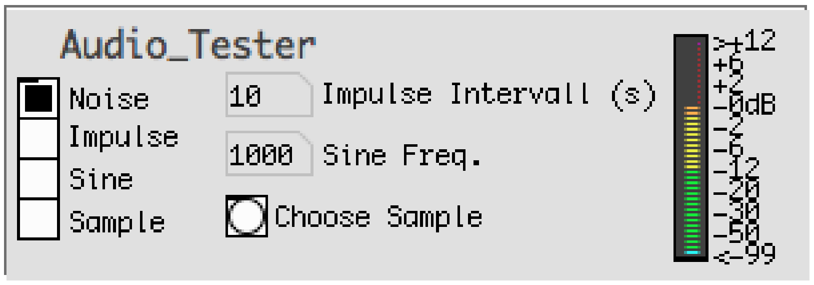


Figure 1.22.: audioTester.pd, zu bauen als Hausübung

2. Modulation and Convolution

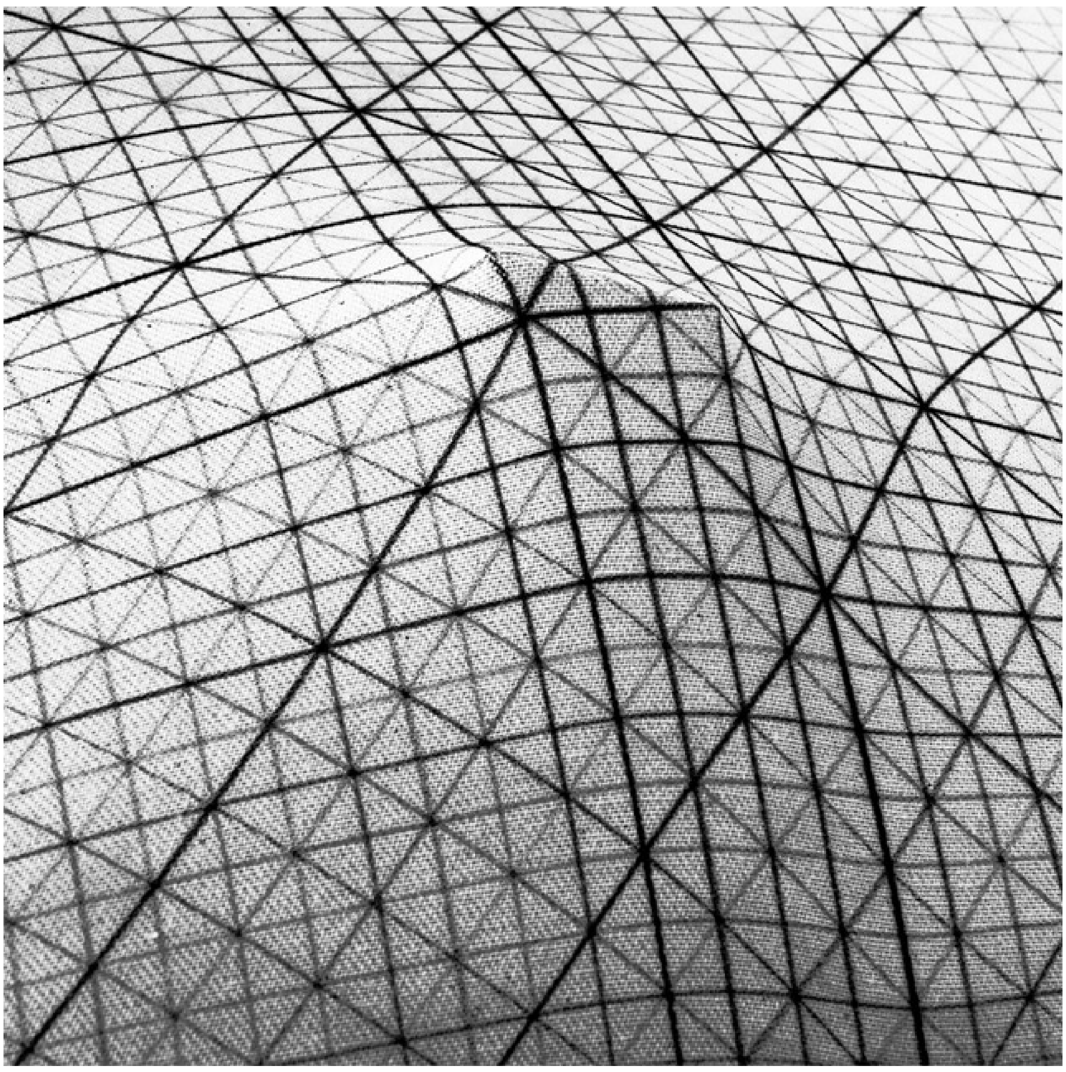


Figure 2.1.: “Surface Modulation” by Richard Sweeney

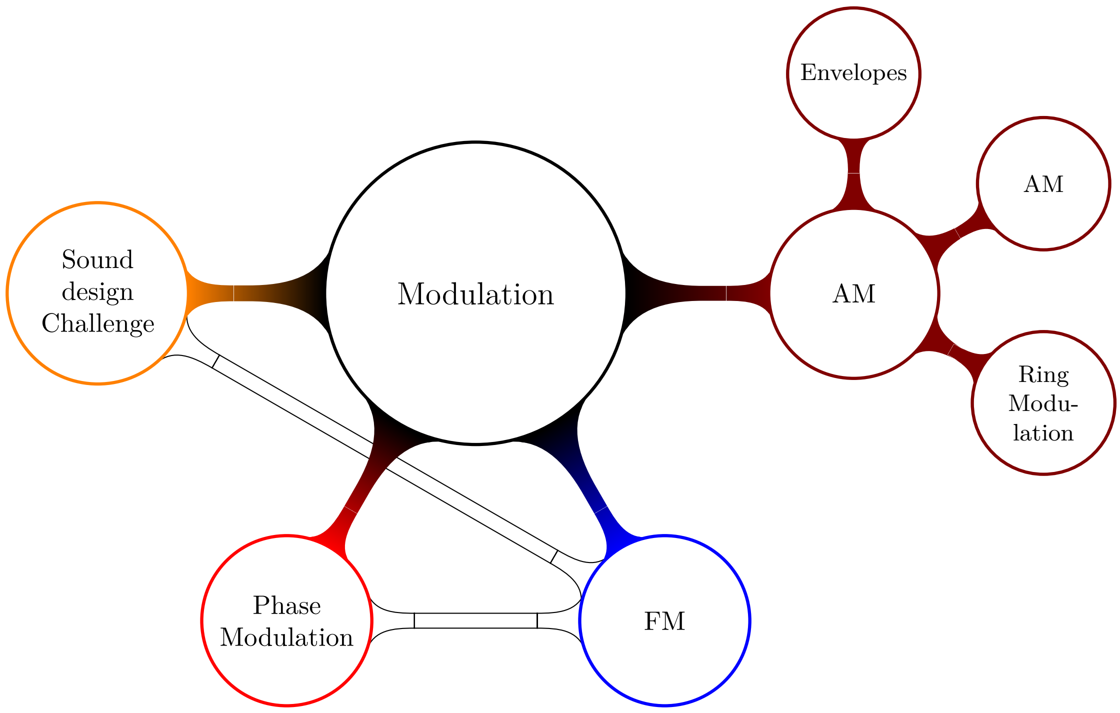


Figure 2.2.: Lecture Contents

2.1. AM

In this section we will try amplitude modulation. Amplitude modulation is used for radio communication (so you'll need to understand this as a technician, modulation techniques are extremely important and this is maybe the simplest one) but also in sound design. We will try to understand the problem from different perspectives at once:

- Doing some math
- listening to it
- brining it in context to beating waves
- seeing it as convolution in the frequency domain

Amplitude modulation means modulating the amplitude of a signal(surprise!). Modulating means changing over time by another signal. So we have some signal, say, a sine wave, and change its amplitude with another signal, say, another sine wave. Talking this concept and reducing it radically, we end up with figure 2.3.

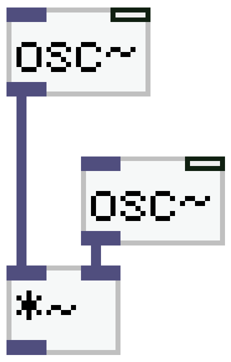


Figure 2.3.: Simplest form of “Ring modulation”.

Of course, we get no sound in figure 2.3 because the frequencies are not initialized, but it shows the general principle. The caption of that figure says “Ring Modulation”. Let’s quickly get some vocabulary straight:

- “Amplitude modulation” might mean any modulation of amplitude
- “Ring Modulation” means *bi-polar* amplitude modulation.
- In sound design, “Amplitude Modulation” might specifically mean unipolar amplitude modulation.

And some more vocabulary to put what we are doing in a musical context:

- Modulating the amplitude is called “Tremolo” in music [1]
- Modulating the pitch or frequency (“FM”) is called “vibrato” in a musical context.[2]

Enough words, let’s look at what AM looks like, look at figure 2.4.

Question 2 *If we would listen to the signal depicted in the bottom plot of figure 2.4, what do you think we would hear? Try to imagine! If you can’t, use pure data to test it! That’s why we are using pd.*

Answer 2 *We would hear a 30Hz sine wave repeatedly rising and falling in amplitude.*

[1]sadly, the fender stratocaster’s “Tremolo Arm” is used to control the pitch. Ignore Fender, they got it wrong. You can trust that most guitar players are confused because of this.

[2]Maybe think about it like this: The F in FM is a bit like the v in vibrato. Just to avoid confusion..

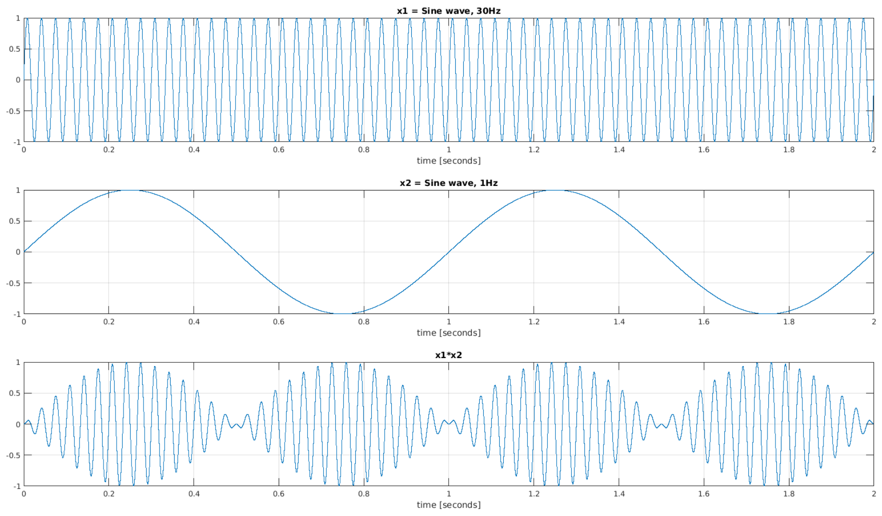


Figure 2.4.: Looking at AM in the time domain

Question 3 *Next question, same plot, same signal. So hopefully you found out that we hear a 30 Hz sine with rising and falling amplitude. At what frequency does the amplitude rise and fall? Remember we are modulating with a 1 Hz sine.*

Answer 3 *Two Hertz.*

Maybe you remember from the waveshaping chapter that we actually calculated what frequencies should come out of AM. Also maybe you remember that sum and difference frequencies should come out, but here we still hear the 30 Hz, just getting louder and softer, so what's wrong? Let's look at the formulas again, just for reference:

$$cos(a) \cdot cos(b) = \frac{cos(a+b) + cos(a-b)}{2} \tag{2.1}$$

So, let's calculate this. We have two oscillators, $x_1(t) = cos(30t2\pi)$ and $x_2(t) = cos(t2\pi)$. We multiply them, ending up with:

$$y(t) = cos(30t2\pi) \cdot cos(t2\pi) \tag{2.2}$$

Ok, the above formula tells us this means:

$$y(t) = \frac{cos(30t2\pi + t2\pi) + cos(30t2\pi - t2\pi)}{2} \tag{2.3}$$

We can now simplify to:

$$y(t) = \frac{cos((30+1)t2\pi) + cos((30-1)t2\pi)}{2} = \frac{cos(31t2\pi) + cos(29t2\pi)}{2} \tag{2.4}$$

Hm, so we get out a 31Hz and 29Hz oscillator. What about that rising and falling in amplitude that we hear *and* observe in the plot, surely there must be something wrong! Do you have a solution to this?

Let's use pure data to help us understand. Make two oscillators, one with 29Hz and one with 30Hz, what do you hear?

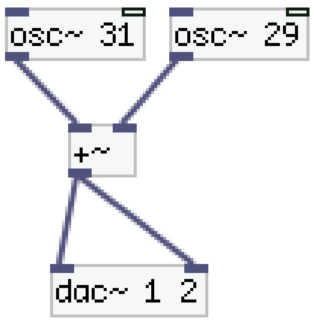


Figure 2.5.: Adding two oscillators with frequencies very close to each other

If you listen to what is depicted in figure 2.5, you will in fact hear the same as if you do the 30Hz /1 Hz amplitude modulation (which indicates that our formula above is correct). What we hear is a phenomenon called beating. The phase of the two oscillators is canceling each other out at regular intervals because the frequencies are so close to each other. In fact the frequency of the beating f_{beat} is always

$$fbeat = |f_1 - f_2| \tag{2.5}$$

So the difference of the two frequencies.

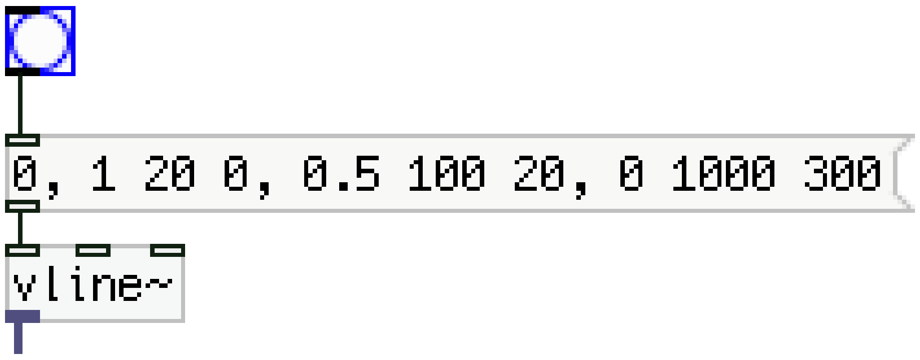


Figure 2.6.: caption

Multiplying two signals in the time domain is equvalent to convolution in the frequency domain and vice versa.
Amplitude Modulation. „tremolo“
Ringmodulation = Bipolar, AM = Unipolar

2.2. FM

Frequency modulation can be used to modulate the frequency, meaning to vary the pitch of a sound, see figure 2.10. But it can also be used to generate overtones and an overall richer spectrum, see figure 2.11. The two “different versions” only differ from each other by different parameter[3] values being used. We can think of Frequency Modulation (FM) as something of the form:

$$y(t) = cos(cos(b)) \tag{2.6}$$

This really is a simplification, but the overall structure of the formula is correct. Looking at figure 2.7 we can see the very basic idea implemented in pure data.

[3]We see which parameters can be adjusted in a minute. But if you want to know them now: modulation frequency, modulation amount and carrier frequency.

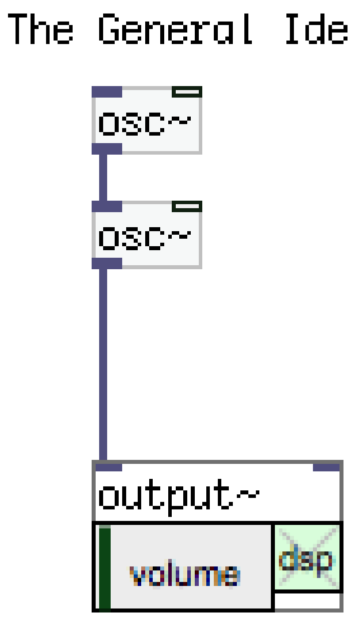


Figure 2.7.: The General Idea of FM

Since the frequencies of the oscillators are not set, we won't hear anything when building this patcher. But looking at it may reveal that the idea simply is to control the frequency of an oscillator using another oscillator.
We can expand the patcher by adding some math to make it more usable, as in figure 2.8.
Naive parameters are f_c (Carrier Frequency), f_m (Modulator Frequency), and A_m (modulation Amount).

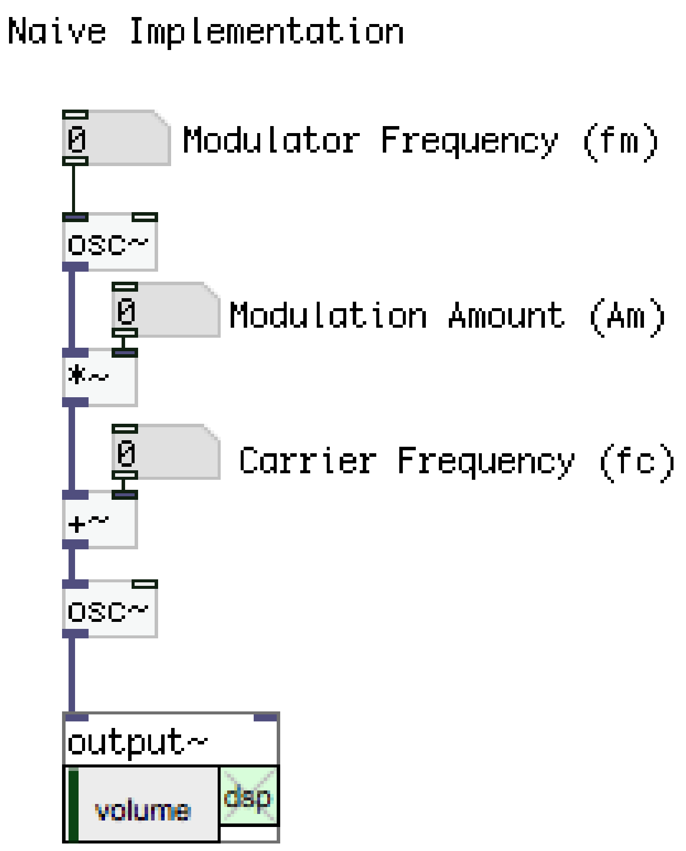


Figure 2.8.: Naive Implementation with Direct Parametrization.

The output frequencies will be

$$f_c \pm n \cdot f_m \tag{2.7}$$

Typically, FM is controlled via *Index*, *Ratio*, and fundamental Frequency. The Index, I is given by Modulation Depth and Modulator Frequency.

$$I = \frac{A_m}{f_m} \tag{2.8}$$

A more controllable Implementation will generate the naive parameters from a Ratio, R, the Carrier Frequency and the Index:

$$f_m = \frac{f_c}{R} \tag{2.9}$$

$$A_m = \frac{I}{f_m} \tag{2.10}$$

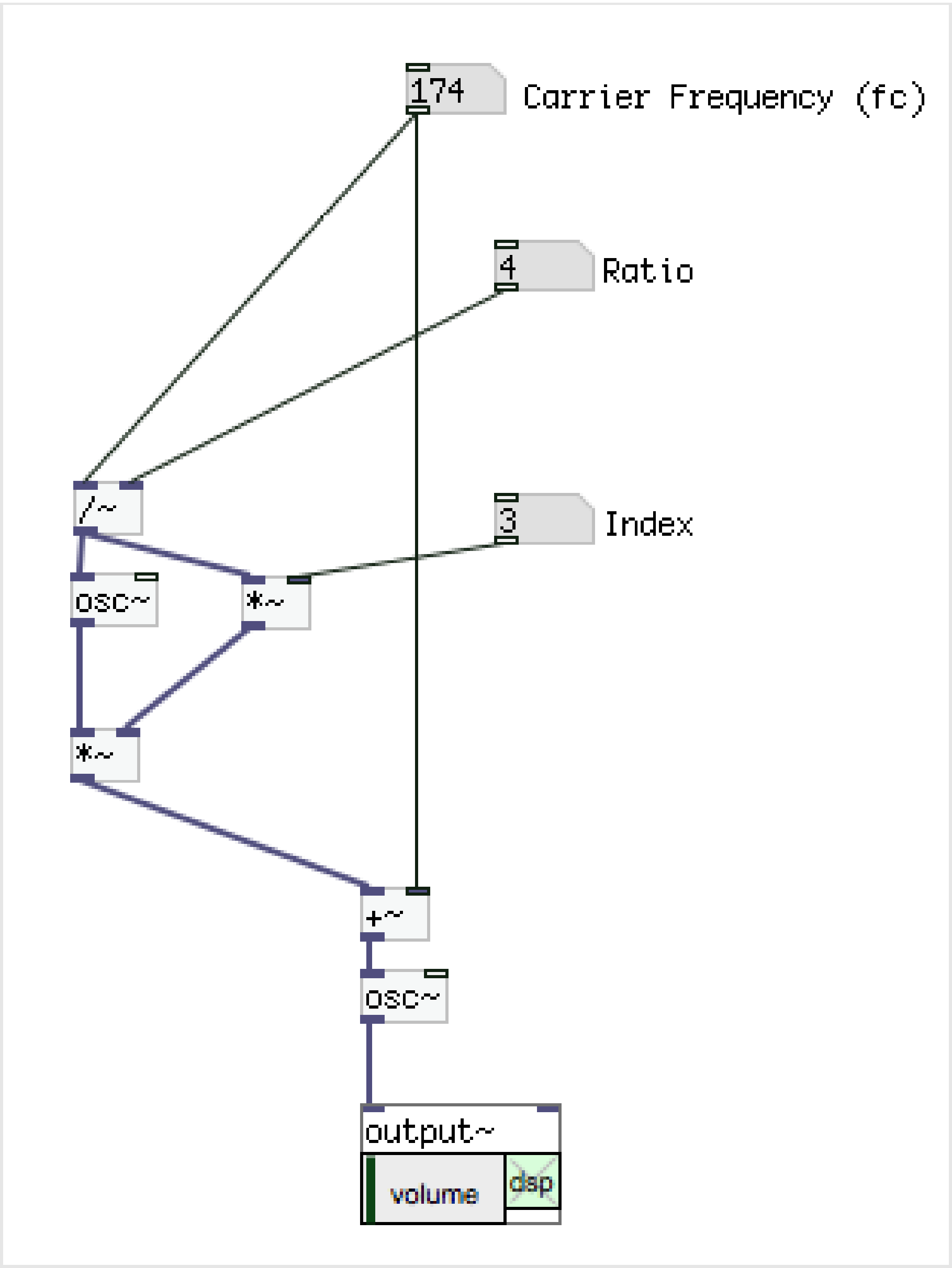


Figure 2.9.: FM with Index and Ratio

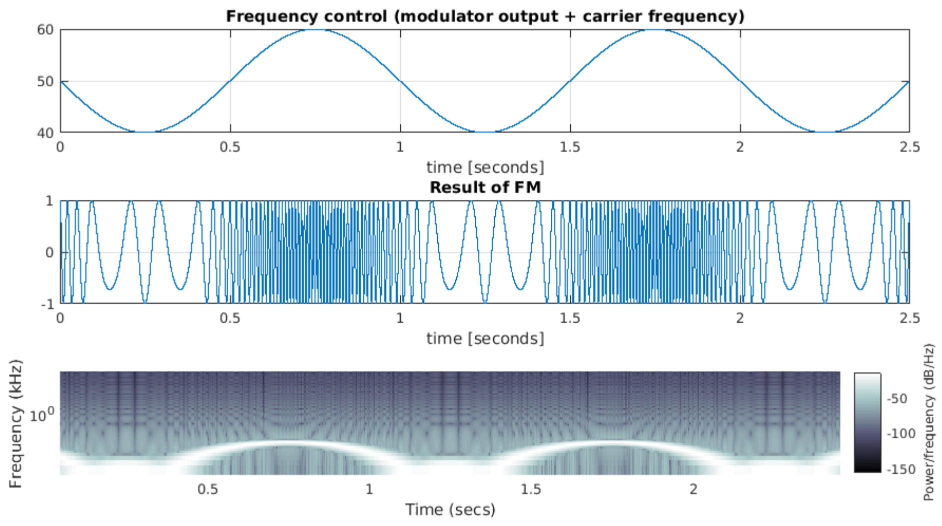


Figure 2.10.: Modulator frequency: 1Hz, Crarrier Frequency 50 Hz, Modulation Amount 10. Please ignore the labeling of the Y axis of the spectrum plot(10^0). It is wrong. The y axis goes from 0 to nyquist=22050Hz.

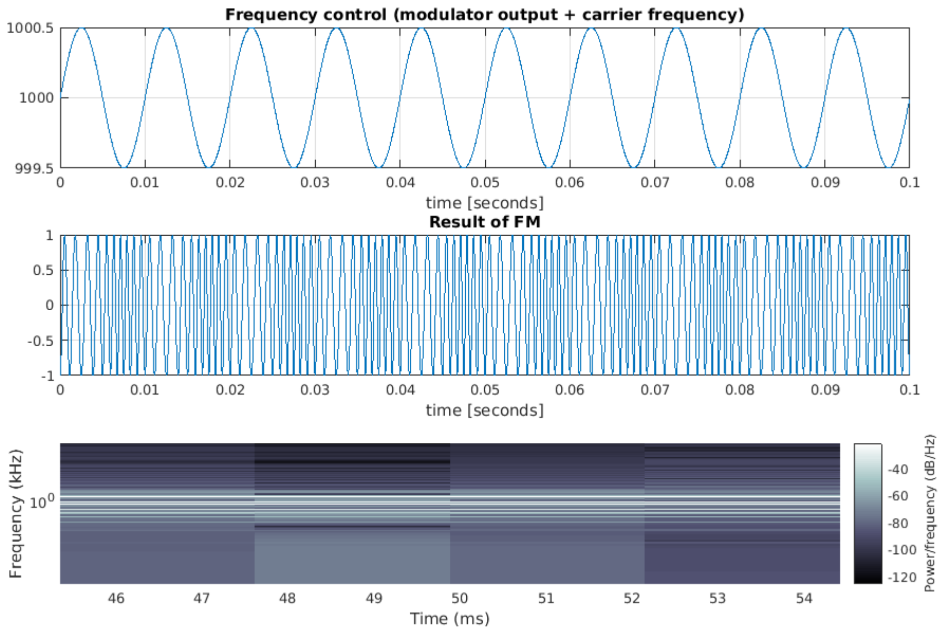


Figure 2.11.: Modulator frequency: 100Hz, Crarrier Frequency 1000 Hz, Modulation Amount 0.5. Please ignore the labeling of the Y axis of the spectrum plot(10^0). It is wrong. The y axis goes from 0 to nyquist=22050Hz.

List of Figures

Bibliography

Farnell, A. (2010). *Designing sound.* MIT Press, Cambridge, Mass.