Mathematics for automated audio system testing

August 2, 2015

J.L.

Many electronic systems, such as computers, TV's and mobile phones have an audio interface, which in these days is implemented as an embedded system containing hardware and software. Typically programming errors in software side may cause problems in audio output in any device.

The final acceptance testing is almost always done manually using measurements in audio lab accompanied with listening tests for true user experience.

To find problems in the early phases of the device development, some parts of the audio quality testing can be automated to be run for example every night. In the optimal situation this partial test automation will save some time from the manual testing effort.

1 Test bench examples

To test the complete system or only the software?

If it is desired to test the complete system, one needs to choose the piece of hardware which handles the recording of the audio output. Typically a high-class computer sound card will be sufficient recording device. If the device has a headphone connection, then audio can be recorded through the line-out of the device. Other option is to build an acoustically isolated test room or a smaller scale test box, and use an external microphone to record the audio. For acoustic measurements a cheap electret microphone has proven to be sufficient.

If only software needs to be tested, then save the audio data in the device itself with some custom test software, and analyse the saved audio clips later using the methods listed in the following sections.

2 Correlation calculations

This seems to be the primary method to verify audio integrity, typically not considered suitable for quality verification. A practical implementation of this method uses a pre-recorded audio sample, which is proven to be faultless. This pre-recording is used as a reference sample. Then the correlation against the reference is evaluated in each test run, where a new test sample is recorded each time. For faulty parts the evaluated correlation drops significantly. The accuracy of the analysis can be controlled by adjusting the

sub-sample size, for which each correlation value is evaluated. So the reference sample is sub-sampled for a certain amount of bits and these bits are correlated against the sample to be tested. Figure 1 gives the preliminary idea of the method. When moving the sub-sample of the reference against a portion of the test sample, the most similar part will give the maximum value of correlation. Obviously the sub-sample from the test sample needs to be larger than the sub sample from the reference, so that a sifting alignment is possible to find the best correlation value. This way it is also possible to find out the time lag between the samples. After some maths of the correlation calculations have been introduced, a practical use of this method will be described.

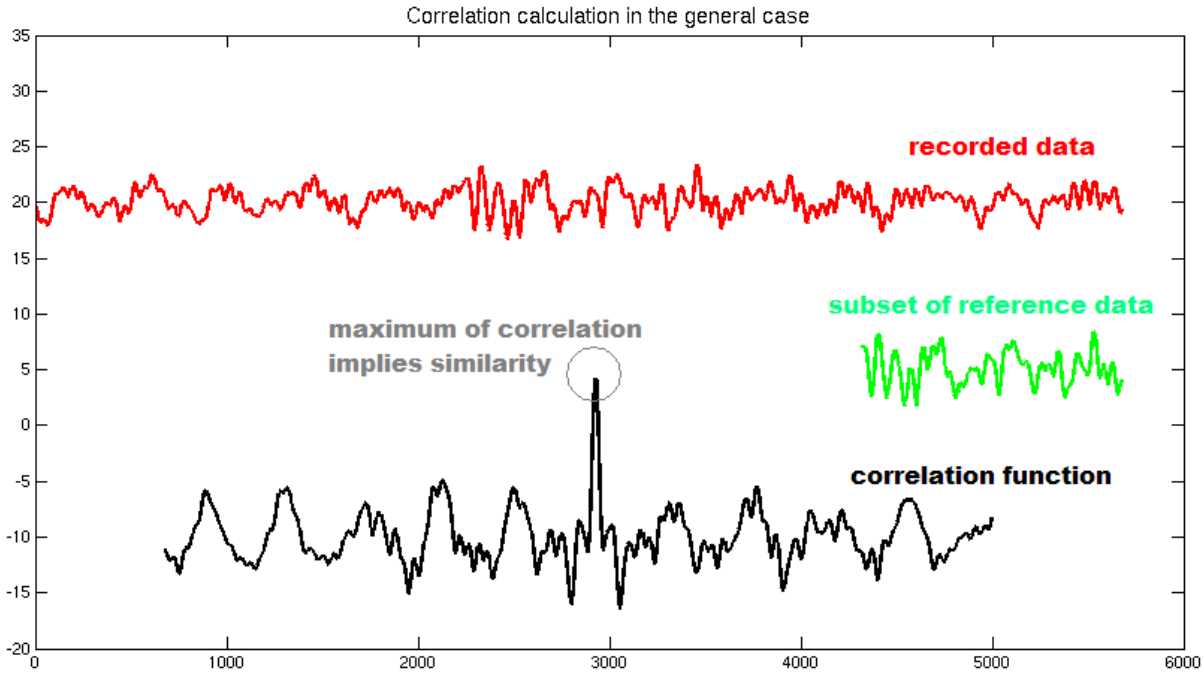


Figure 1: Cross correlation function for real audio signal with reference part

The comparison of two data series, say f and g is implemented using the cross correlation function $f \star g$, that is usually defined as

$$(f \star g)_n = \sum_{m=-\infty}^{\infty} f_m^* \cdot g_{n+m}. \tag{1}$$

The cross correlation function is a set of data points, which give a measure of similarity between two data sets and the index of a certain cross correlation function value can be used to determine the delay between points of two

discrete data sets. Here f can be selected to depict the reference file and g is then the recorded data. As the cross correlation function gives a measure of similarity, a reference value is still needed to give a comparable value to determine the actual similarity of the two signals. This reference value is obtained by calculating the autocorrelation $f \star f$ of a discrete data set. Figure 2 depicts the autocorrelation function of a simple sine wave.

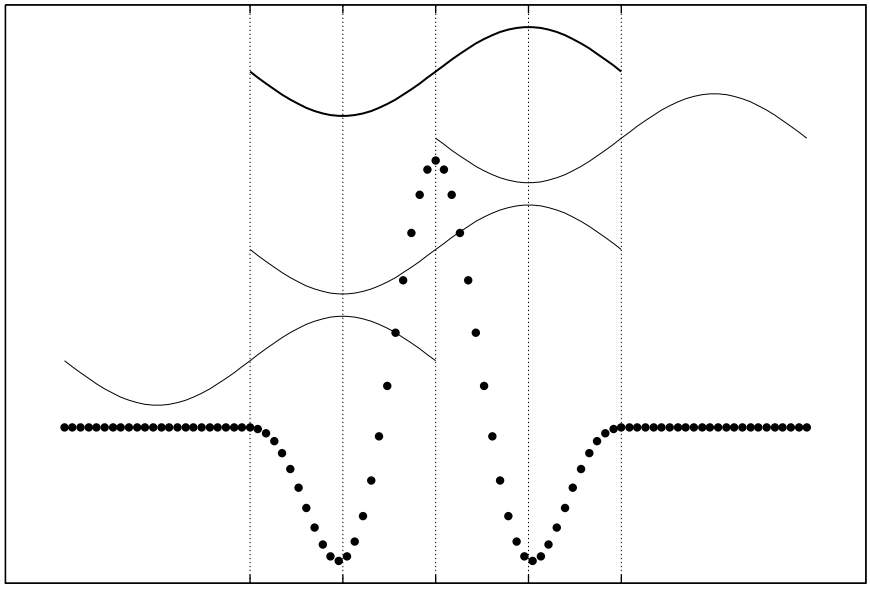


Figure 2: Autocorrelation function of a simple sine wave

In practise we do not have infinite amount of data points in our data series so that the cross correlation equation (1) could be determined better this way:

$$(f \star g)_n = \sum_{m=-K}^{K} f_m^* \cdot g_{n+m}, \tag{2}$$

where K is some constant that defines the frame size $F_S \in [-K, K]$ from the reference file point of view.

So by calculating the autocorrelation from the reference signal and comparing it to the cross correlation values calculated from some closed interval $[-K, K]$, the similarity is found when the difference between autocorrelation value and some specific cross correlation value reaches some kind of minimum. It is also assumed that where the cross correlation has a maximum value, it indicates the point where the two data sets are most similar in comparison. Based on these two facts, the difference in similarity can be

then obtained, e.g. by calculating the relative difference between the auto-correlation and cross correlation maximum values:

$$\Delta X = \frac{f \star f - max(f \star g)}{f \star f}, \tag{3}$$

The most critical part of this correlation calculation is the primary alignment of the two data sets. For this the general definition of cross correlation function in equation (1) can be used assuming that the maximum value of this function finds the correct alignment of the two data sets. Because now we are connected with the theories of statistics and probabilities, the more data points are taken to the calculation of the correlation, the more likely it is to find the correct alignment point of the two data sets. But please note that this is not a fool proof method of finding the correct alignment in any case. Probabilities can be very tricky! The proper alignment is found most easily for randomly varying signals. For continuously repeating signals, such as a sine wave, the proper alignment cannot be found without some extra checks.

The calculation procedure for the cross correlation from equation (1) is quite simple, but it takes a lot of computing resources because of the large amount of multiplications done during the calculation. There exists a connection between multiplication in the frequency domain and correlation in the time domain. The correlation calculation for large data sets can be made much more faster by first applying a Fast Fourier Transform to the data sets and doing a multiplication in the frequency domain:

$$\begin{aligned} \mathcal{F}(f * g) &= \mathcal{F}(f)\mathcal{F}(g), \\ \mathcal{F}(f * g) &= F_k \cdot G_k \end{aligned} \tag{4}$$

The equation (4) implies that the multiplication in the frequency domain is done component by component and when taking the inverse FFT from the multiplication result, we have the original correlation function:

$$(f * g) = \mathcal{F}^{-1}\left[\mathcal{F}(f)\mathcal{F}(g)\right]. \tag{5}$$

Few questions arise... How can we here determine the different values of correlation for some interval $[-K, K]$ when both of the data series must have the same amount of data points when doing the FFT calculation? The correlation via FFT should be understood as a *circular* correlation where the reference signal is "moved" along the recorded data points so that the last value in the reference frame jumps to the first position in the frame. The idea

of multiplication is otherwise exactly the same as in the normal correlation calculation. Let's take a little example, where x is a frame from the reference data set and y is part of the acquired data set:

$$x = \{2, 3, 0, 0\} \quad ; \quad y = \{1, 2, -3, 5, 1, 2, -3, 5\}$$

The cross correlation function between these is by equation (1):

$$(x \star y) = \{8, 0, 9, 13, 8\},$$

where $n \in [-2, 2]$. Now let's reduce the frame y to have the same amount of data points as x. Of course we need to also make sure that the amount of data points is suitable for the FFT in general, i.e. some exponent of 2:

$$x = \{2, 3, 0, 0\} \quad ; \quad y = \{1, 2, -3, 5\}.$$

After FFT, multiplication in the frequency domain, and inverse FFT, we get:

$$(x \star y) = \{8, 0, 9, 13\},$$

so that the length of the cross correlation function $(x \star y)$ is the same as the data sets under examination. Here the indices will in general start from 0, but please note that the first 4 values of the cross correlation function are the same as the result via the normal correlation multiplication. Also note that the recorded samples are extended to be periodic in the first calculation. This depicts the periodical behaviour and the cyclic nature of the cross correlation calculation in the frequency domain via FFT.

The point here is that both of the methods give the same results if the reference data set is smaller than the recorded set, so that the periodicity effects of the FFT calculation can be worked around. The reference data is here padded with zeros to make the example more clear. In the FFT correlation scheme the first index of the correlation function represent the index where the two data series are perfectly aligned, then the following indices' values correspond to the delay of the recorded signal with respect to the reference signal. When the indices reach the point where only $N - N_{ref}$ indices remain, these inform if the reference signal is delayed with respect to the recorded signal. Unfortunately this does not tell the truth about the delay of the reference, because we don't have the real information about the behaviour of the recorded signal outside the frame of reference. The FFT just assumes the signal to be periodic.

After the preliminary matching of the signals has succeeded, we can use the normal time domain multiplication to determine the correlation and just take slightly bigger frame from the recorded signal to follow the possible alignment change during the correlation calculation. This makes the calculation procedure a bit faster.

The correlation function as defined by equation (1) is quite prone to errors. Lots of problems arise for example from such a common thing as an amplitude difference between the two signals to be compared. In the general case when multiplying two numbers, say x and y, with errors in them, we have:

$$x \cdot y = (x + \Delta x) \cdot (y + \Delta y) = x \cdot y + x \cdot \Delta y + y \cdot \Delta x + \Delta x \cdot \Delta y \tag{6}$$

Therefore the error introduced in the correlation calculation is expressed as

$$\Delta(f \star g)_n = \sum_{m=-\infty}^{\infty} g_{n+m} \cdot \Delta f_m^* + f_m^* \cdot \Delta g_{n+m} + \Delta f_m^* \cdot \Delta g_{n+m}, \tag{7}$$

but in this case when the reference data is naturally expected to be constant, equation (7) reduces to

$$\Delta(f \star g)_n = \sum_{m=-\infty}^{\infty} f_m^* \cdot \Delta g_{n+m}. \tag{8}$$

Since we are expecting the two data sets to have some similarity, perhaps the best way is to express the whole cross correlation function (1) with error added with respect to the other data set, so that the difference between the two data sets is seen as an error Δf_{n+m} in the other data set:

$$(f \star g)_n = \sum_{m=-\infty}^{\infty} f_m^* \cdot g_{n+m} = f_m^* \cdot (f_{n+m} - \Delta f_{n+m}) \tag{9}$$

When performing the multiplication in (9) we have

$$(f \star g)_n = \sum_{m=-\infty}^{\infty} f_m^* \cdot f_{n+m} - f_m^* \cdot \Delta f_{n+m} \tag{10}$$

and by expanding with f_{n+m}:

$$\begin{aligned} (f \star g)_n &= \sum_{m=-\infty}^{\infty} f_m^* \cdot f_{n+m} - f_m^* \cdot \frac{f_{n+m} \cdot \Delta f_{n+m}}{f_{n+m}} \\ &= \sum_{m=-\infty}^{\infty} f_m^* \cdot f_{n+m} \cdot \left(1 - \frac{\Delta f_{n+m}}{f_{n+m}}\right) \end{aligned} \tag{11}$$

Equation (11) will help us to solve the error differences that come from constant amplitude difference between the two compared signals. Assuming that data sets f and g are otherwise similar, but have a constant amplitude difference, then the fraction $\frac{f_n}{g_n}$ gives the same constant value at every point n in the data series. This fraction corresponds to the term $\frac{\Delta f_{n+m}}{f_{n+m}}$ in equation (11), which becomes therefore a constant value expression and when inserting the results from equation (11) to the equation (3), we get:

$$\begin{aligned} \Delta X &= \frac{\sum f_m^* \cdot f_{n+m} - \sum f_m^* \cdot f_{n+m} \cdot \left(1 - \frac{\Delta f_{n+m}}{f_{n+m}}\right)}{\sum f_m^* \cdot f_{n+m}} \\ &= \frac{\Delta f_{n+m}}{f_{n+m}}, \end{aligned} \tag{12}$$

which is equivalent to the relative error term of the reference signal.

Luckily there is a workaround for neglecting the amplitude difference. What if one would consider the compared data arrays to be vectors? The correlation function (1) looks quite similar to the dot product calculation between two vectors. And indeed, in practise the cross correlation equation (1) corresponds the dot product:

$$\vec{a} \bullet \vec{b} = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n = \sum_{i=1}^{n} a_i b_i = a \star b. \tag{13}$$

Vectors have a very unique property, which applies for all dimensions: normalisation. Vector magnitude is defined as

$$|\vec{a}| = \sqrt{\vec{a} \bullet \vec{a}} = \sqrt{a_1 a_1 + a_2 a_2 + \ldots + a_n a_n}. \tag{14}$$

When dividing the original vector $\vec{a}$ with the lenght $|\vec{a}|$ (i.e. magnitude), we have a unit vector

$$\hat{a} = \frac{\vec{a}}{|\vec{a}|}. \tag{15}$$

When extracting the magnitude as a separate entity and using only the unit vectors for calculating the correlation, the cross correlation function $\hat{a} \star \hat{b}$ will not be sensitive to amplitude differences, it tests only the similarity of the signals and does not care about the magnitude. If the amplitudes need to be tested equal, then the magnitudes can be used for asserting that.

Unfortunately (at least to my knowledge) the cross correlation calculated via FFT cannot be normalised, so the correlation calculation using normalised vectors need to be done using normal multiplication process.

So based on the information presented above, the practical implementation of a cross correlation testing for audio signals should be done as follows. Prepare the reference audio sample so that the recorded test audio sample always lags the reference when starting the correlation analysis. Use the Fast Fourier Transform *only once* to find the maximum correlation value from a relatively long subset of the test sample. This will reveal the primary lag value of the signals and it can be used for making a primary lag adjustment for the reference and the test samples.

Because Fast Fourier Transform is used to calculate the correlation the size taken from the recorded signal must be a power of 2, e.g. 262144 samples. The amount of samples taken from the reference data should be large enough to get a reliable correlation result, but must be also lot smaller than the amount of samples taken from the recorded test signal. The effective number of cross correlation values is $N - N_{ref}$, where N is the amount of samples taken from the recorded test signal and N_{ref} the amount of samples from the reference data respectively. Empiric observations have proven that about 50 000 samples taken from the reference signal gives quite reliable correlation results. When using e.g. 40 kHz sampling frequency, we take about one second subset from the reference file and the largest possible lag in the recorded data can be about 5 seconds when 262144 samples is taken to the analysis.

When the primary alignment of the signals has been made, the correlation calculation proceeds in the file by taking some predetermined frame size of data from the reference file and compares this to a frame of samples taken from the recorded test file. The frame size for the recorded file is a little larger than the reference file frame size, because this way one can follow the possible change in delay between the two signals over time. In practice the samples read from the recorded test file is < 100 samples more compared to the reference data subset. Note that in this phase we are using the vector multiplication. This process can be made lightning fast if we take only 6 or 8 samples more from the recorder test file compared to the reference. The lag should be followed all the time and if it seems that the lag growing to be more than ± 3 samples, then make a new lag correction and continue the vector multiplication.

Defects found using the correlation method are typically small breaks in the audio stream as shown in Figure 3. These are easy verify later on from the

recorded audio file.

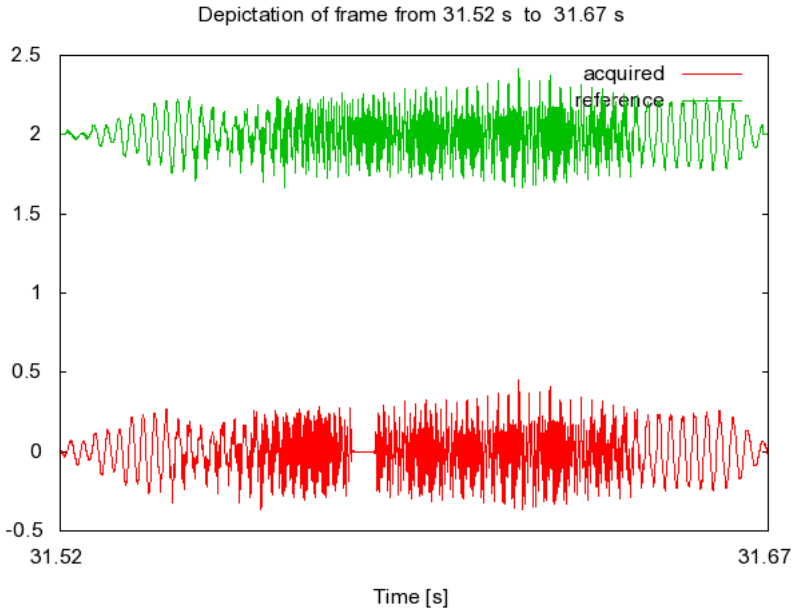


Figure 3: A break in the recorded test file

Other small differences are not so evident, and they typically need some more investigations of the true root cause. Obviously there is something different in the signals shown in Figure 4...

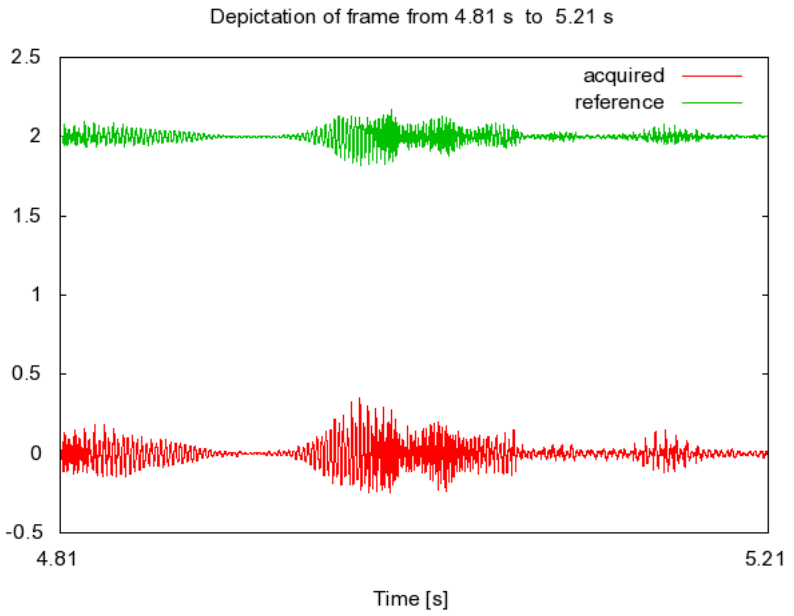


Figure 4: Slight differences between the reference and the recorded test file

3 Derivative calculations

The basis for the derivative analysis is to use sinusoidal test signals. The derivative of a sine wave is a cosine wave and vice versa. The basic difference between sine and cosine is only 90 degrees phase difference. The maximum value of a sinusoidal derivative is easily determined, so when the signal is pure sine it should be easy to verify against the maximum value that there is no extra disturbances in the signal. The derivative analysis is intended to reveal fast unexpected disturbances in the signal, like snaps and pops.

Numerical mathematical methods define the difference $\Delta f(x)$ between two consecutive data points

$$\Delta f(x) = f(x+k) - f(x). \tag{16}$$

This equation (16) is known as the forward difference formula. Naturally there exists a backward difference formula

$$\nabla f(x) = f(x) - f(x-k), \tag{17}$$

and the central difference equation:

$$\Delta f(x) + \nabla f(x) = f(x+k) - f(x-k). \tag{18}$$

These formulas can be extended as numerical derivatives when bringing in the slope property of the derivative. Then the relative change in some variable, call it x, can be determined via difference formulas.

$$\frac{\Delta f(x)}{\Delta x} = \frac{f(x+k) - f(x)}{\Delta x} = \frac{f(x+k) - f(x)}{x+k-x} = \frac{f(x+k) - f(x)}{k}. \tag{19}$$

When combining the forward and backward difference formulas, a derivative based on the central difference is obtained.

$$\begin{aligned} \frac{\Delta f(x) + \nabla f(x)}{\Delta x} &= \frac{f(x+k) - f(x) + f(x) - f(x-k)}{\Delta x} \\ &= \frac{f(x+k) - f(x-k)}{x+k-(x-k)} \\ &= \frac{f(x+k) - f(x-k)}{2k}. \end{aligned} \tag{20}$$

All of these numerical derivative equations are officially derived from the Taylor series expansion by taking the first terms of the series, making the proper substitutions and subtractions.

The amplitude and the fundamental frequency need to be determined before the derivative analysis can work effectively. The fundamental frequency f_0 can be determined by following the signal zero crossings and using the equation

$$f_0 = \frac{f_s}{N}, \tag{21}$$

to get the actual fundamental frequency of the signal. The notation f_s means the sampling frequency, and N is the amount of samples between two consecutive zero crossings. Of course it is expected that the measured signal is clean as possible so that the fundamental frequency would be easy to determine.

The amplitude of the signal can be determined by e.g. summing the absolute values of minimum and maximum values, taking the average and dividing by two. Much better option would be to calculate the amplitude of each period by Discrete Fourier Transform. Then take the average of all calculated amplitudes. Again, the cleaner the measured signal is, the more accurate the result is.

Without worrying too much about strict mathematical formalism, the differential of the sine wave can be calculated as:

$$d[A \cdot \sin(\omega t)] = \frac{A \cdot \sin(\omega[t + dt]) - A \cdot \sin(\omega t)}{dt} \cdot dt = A\omega \cdot \cos(\omega t)dt,$$

where

$$\omega = 2\pi f_0 = 2\pi \frac{f_s}{N},$$

and now because now we are not handling the infinitesimal differentials, we can use a notation change $dt \longmapsto \Delta t$, where the Δt in this context is the reciprocal value of sampling frequency. Now we still want to normalise the derivative to a unity value, which $\forall t \in R[-1, 1]$. That can be done like this:

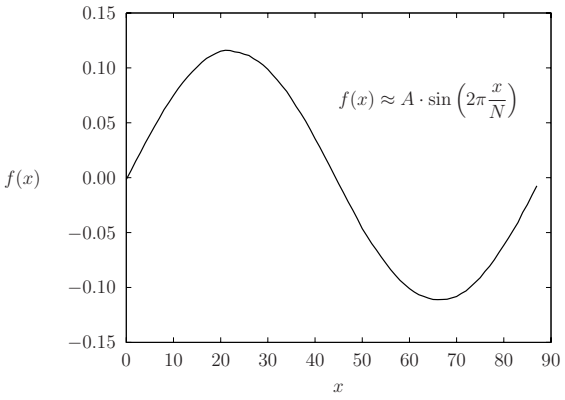
$$A\omega \cdot \cos(\omega t)\Delta t \quad \Big| : A\omega\Delta t$$

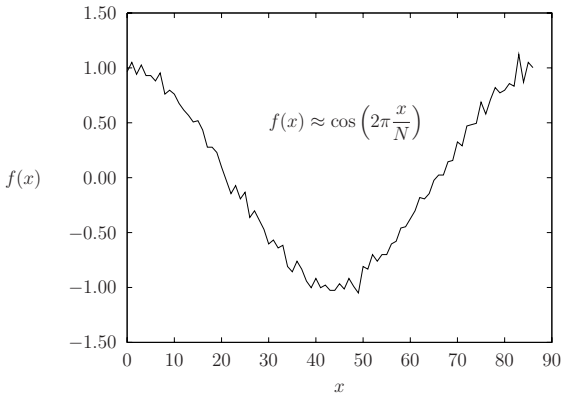
$$\Rightarrow \cos(\omega t)$$

To get the differential of a basic sine wave $A \cdot \sin(\omega t)$ normalised as a cosine wave with unity amplitude, the differential needs to be divided with a factor $A\omega\Delta t$. In this case we would get

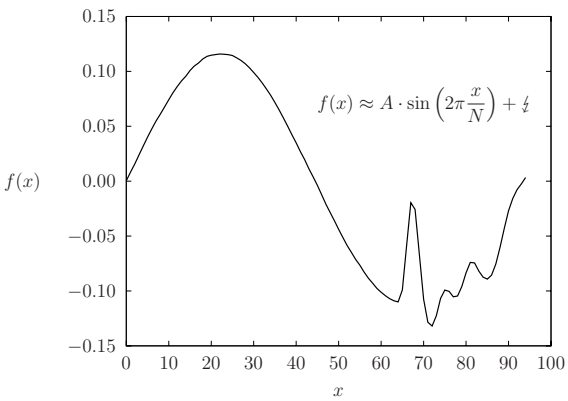
$$\frac{A\sin(\omega[t+\Delta t]) - A\sin(\omega t)}{A\omega\Delta t} = \frac{A\sin(\omega[t+\Delta t]) - A\sin(\omega t)}{A\frac{2\pi}{N}} = \cos(\omega t).$$

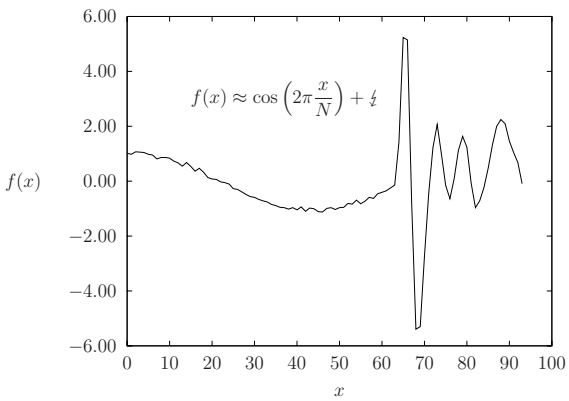
This way the value of the derivative of the test signal will stay between the interval [-1, 1] and if the signal has some changes faster than the fundamental frequency, those will be detected as values larger than $|1|$. In practical testing one can then use some constant value as a limit for a failing case. The following two figures show a clean sine and its derivative, which is normalised to have unity amplitude.





Then the following two figures show a defected sine wave and its derivative. From here it is easy to assert that the amplitude of the derivative signal is over 1.





4 Discrete Fourier Transform

The DFT calculation can be used in audio testing, especially when the test signal is analytic and composed of several harmonic sine waves, whose amplitude relation is known beforehand. Then the DFT can be used to evaluate the amplitudes of each component and make assertions accordingly.

The fundamental idea of the whole Fourier analysis is that any periodic function $f(x, ...)$ can be modelled with sums of sines and cosines of harmonic frequencies

$$f(x) = a_0 + \sum_{n=1}^{\infty}(a_n \cos nx + b_n \sin nx). \tag{22}$$

The idea of Fourier series extends also to a set of transforms both continuous and discrete time. The discrete Fourier Transform (DFT) X_k of data series x_n is usually defined as:

$$X_k = \frac{1}{N}\sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi k \frac{n}{N}}, \tag{23}$$

where N is the total amount of the numbers in the data sequence x_n. In this version there exists also a normalisation factor $\frac{1}{N}$ in front, but usually this is connected to the inverse DFT. This normalisation coefficient is essential if one wants to dig out the true amplitudes of the harmonic frequency components from the data series x_n. With the help of Euler equations

$$\begin{aligned} e^{inx} &= \cos nx + i \sin nx \\ e^{-inx} &= \cos nx - i \sin nx \end{aligned} \tag{24}$$

The DFT can be written in the form:

$$X_k = \frac{1}{N}\sum_{n=0}^{N-1} x_n \cdot \left[\cos\left(2\pi k \frac{n}{N}\right) - j \sin\left(2\pi k \frac{n}{N}\right)\right] \tag{25}$$

and the equation (25) can still be separated into real and imaginary parts:

$$\begin{aligned} \Re(X_k) &= \frac{1}{N}\sum_{n=0}^{N-1} x_n \cdot \cos\left(2\pi k \frac{n}{N}\right) \\ \Im(X_k) &= \frac{1}{N}\sum_{n=0}^{N-1} -j \cdot x_n \cdot \sin\left(2\pi k \frac{n}{N}\right) \end{aligned} \tag{26}$$

The basic idea behind the DFT is to calculate a correlation between the data series x_n and harmonic frequencies k of the fundamental frequency determined by the sample size N which represents the amount of samples in one

full period of x_n. So the periodicity assumption from the trigonometric series is still there... The possible phase difference is covered by calculating both the sine and cosine correlation, and the result is a squared sum of these two components.

When examining the definition of the inverse DFT:

$$x_k = \sum_{n=0}^{N-1} X_n \cdot e^{i2\pi k \frac{n}{N}}, \tag{27}$$

it can be noticed that the values X_k obtained from the DFT are equivalent to the amplitude coefficients of the complex Fourier series. From the Euler equations (24) also follow the results

$$e^{inx} + e^{-inx} = 2 \cdot \cos nx \quad \Rightarrow \quad \cos nx = \frac{1}{2}\left(e^{inx} + e^{-inx}\right) \tag{28}$$

$$e^{inx} - e^{-inx} = 2 \cdot i \sin nx \quad \Rightarrow \quad \sin nx = \frac{1}{2i}\left(e^{inx} - e^{-inx}\right) \tag{29}$$

and inserting these to the trigonometric Fourier series (22) it follows that

$$\begin{aligned} f(x) &= a_0 + \sum_{n=1}^{\infty}\left(\frac{1}{2}a_n \cdot \left[e^{inx} + e^{-inx}\right] - \frac{1}{2}ib_n \cdot \left[e^{inx} - e^{-inx}\right]\right) \\ &= a_0 + \sum_{n=1}^{\infty}\left(\frac{1}{2}\left[a_n - ib_n\right] \cdot e^{inx} + \frac{1}{2}\left[a_n + ib_n\right] \cdot e^{-inx}\right) \\ &= \sum_{n=-\infty}^{\infty} c_n \cdot e^{inx} \end{aligned} \tag{30}$$

The last form in (30) needs a little explanation. The sum is extended to cover the whole set of integers from $-\infty$ to $+\infty$ and use the notations

$$\begin{aligned} c_n &= \frac{1}{2}\left(a_n - ib_n\right) = X_n \\ c_{-n} &= \frac{1}{2}\left(a_n + ib_n\right) \\ c_0 &= a_0 \end{aligned} \tag{31}$$

This gives a relation to the amplitudes of the complex (c_n) and real (a_n, b_n) forms of the Fourier series. Hence, this implies that from the results of DFT one can determine true amplitudes of the harmonic frequency components of any signal.

In the general case, where the complex amplitudes have both real and imaginary parts, the real amplitude of some harmonic frequency can be determined by calculating the squared sum

$$A_k = 2 \cdot \sqrt{\Re(X_k)^2 + \Im(X_k)^2} = \sqrt{a_k^2 + b_k^2}. \tag{32}$$

Also the phase information is included in the amplitude spectrum and it can be obtained from:

$$\phi_k = \arctan \frac{\Im(X_k)}{\Re(X_k)} \tag{33}$$

The simplest way to prove that the DFT gives the true amplitudes of the harmonic components of a given data set, is to do a simple calculation using the sine wave as the test signal. Suppose that we have only the fundamental frequency of pure sine wave with amplitude A in our system to be measured. We measure through an AD converter exactly one full period of the sine wave with N samples. Then we do the DFT to the fundamental frequency:

$$\begin{aligned}
&\frac{1}{N}\sum_{n=0}^{N-1} A \cdot \sin\left(2\pi\frac{n}{N}\right) \cdot \sin\left(2\pi\frac{n}{N}\right) = \frac{A}{N}\sum_{n=0}^{N-1}\left[\sin\left(2\pi\frac{n}{N}\right)\right]^2 = \\
&\frac{A}{N}\sum_{n=0}^{N-1}\left[\frac{1}{2i}\left(e^{i2\pi\frac{n}{N}} - e^{-i2\pi\frac{n}{N}}\right)\right]^2 = \frac{A}{N}\sum_{n=0}^{N-1} -\frac{1}{4}\left(e^{i4\pi\frac{n}{N}} - 2 + e^{-i4\pi\frac{n}{N}}\right) = \\
&\frac{A}{N}\sum_{n=0}^{N-1}\frac{1}{2} - \frac{1}{2}\cos\left(4\pi\frac{n}{N}\right) = \frac{A}{N}\cdot\frac{N}{2} - \sum_{n=0}^{N-1}\frac{1}{2}\cos\left(4\pi\frac{n}{N}\right) = \frac{A}{2}
\end{aligned} \tag{34}$$

So we get N times the half value of the real amplitude of the sampled sine wave. This N is divided out by the normalisation factor defined in (23). The cosine term in the equation is always zero, when the sampling frequency is an exact multiple of the measured sine wave frequency. Otherwise the cosine term adds the spectral leaking properties to our system because of asymmetric sampling or wrong truncation point of the measured signal. Also, one must be careful to truncate the test signal to contain exactly one full period, otherwise spectral leaking will follow. So in the DFT analysis this can be thought as using a rectangular window function with amplitude 1 to multiply the original signal. The most important point is that we can get the original amplitude of the signal by dividing by N and multiplying with 2.

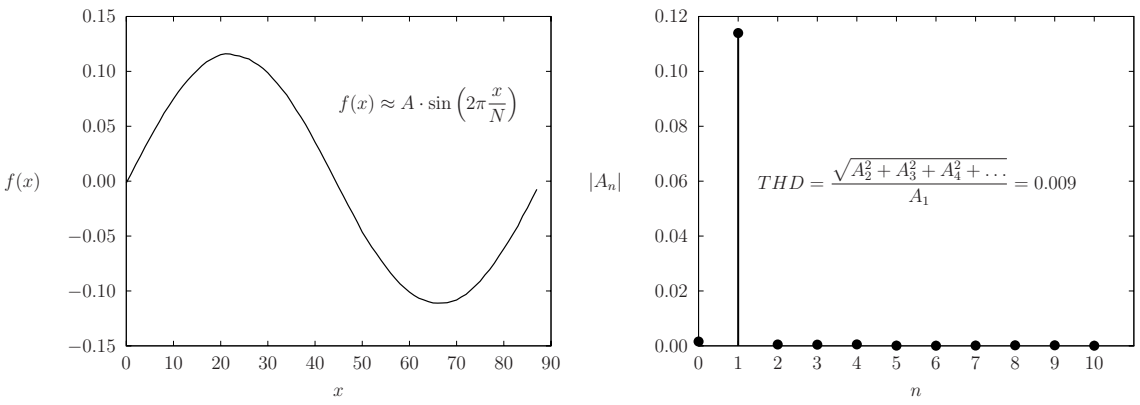
5 Total Harmonic Distortion

Distortion of the signal is quite difficult to define for a general signal that has an undefined number of different frequencies present. For simple test signals such as sine, square, sawtooth etc. the distortion is much more easier to define. The total harmonic distortion compares the amplitudes of the harmonic overtones to the amplitude of the fundamental frequency. There are several different definitions for the THD in different branches of technology areas. In audio frequency calculations the Total Harmonic Distortion is usually defined as:

$$THD = \frac{\sqrt{A_2^2 + A_3^2 + A_4^2 + \cdots + A_n^2}}{A_1}, \tag{35}$$

where the amplitudes of the harmonic frequencies A_n are square-summed ("power domain sum") and divided by the amplitude of the fundamental frequency A_1. The value of THD is commonly expressed as percentage from the amplitude of the fundamental frequency. When using simple test signals where the overtone structure of the signal is theoretically known, it is possible to use the THD to measure the actual distortion i.e. the transient effects of the system under analysis.

The following picture comparison illustrates the THD value together with the DFT analysis as it could be used in locating audio problems in automated testing. Notice how the DFT analysis gives the correct amplitude value for the simple sine wave. The excess THD found from the clean sine wave is coming trough because of the sampling frequency is not a multiple of the tested wave and the truncation point has not been able to isolate on full period of the wave. This is typical and acceptable. The full wave can be extracted automatically for analysis by following the zero crossings of the measured signal. When using a simple sine wave or a set of harmonic sinusoids this will work nicely.



To locate actual errors the THD works pretty well as illustrated below: