

Software Sound Synthesis

Module Objectives:

This module is designed to provide students an understanding of sound synthesis and the use of computer music languages .

Module Content:

This module will explore computer instrument design and classic synthesis techniques, as well as introducing students to newer methods and concepts. The csound language will be taught and used in instrument development and sound synthesis.

Programme

1. **Introduction to digital signals**
2. **Basic concepts of software synthesis:** introducing music programming languages and csound.
3. **Synthesis foundations:** unit generators, oscillators, wavetables, envelopes and signal manipulation.
4. **Additive synthesis techniques.**
5. **Filters and subtractive synthesis.**
6. **Non-linear/distortion synthesis techniques:** Frequency Modulation, Phase Modulation and Waveshaping.
7. **Granular and Formant (FOF) synthesis.**
8. **Introduction to Physical Modelling.**

Assessment

Assessment will consist of:

(a) One software sound synthesis/electronic music composition project, with 01/Dec deadline. Typically, it will involve a 3-minute piece with a specific brief, which will demonstrate the student's ability to handle software sound synthesis.

(b) Written examination in January. This will be based on questions on synthesis techniques, concepts and programming. Exam papers for similar modules taught in previous years can be obtained from the library.

Software Tools

The following software tools will be used in this course:

- (a) **Csound:** software synthesis environment, computer music language and sound compiler. This is free software, which can be downloaded from a number of WWW sites (<http://csounds.com>).
- (b) **Soundfile editing:** any soundfile editing software can be used. The lab facilities provide *soundforge* and *Audacity*.
- (c) **Command tools/text editors:** as we will use Csound as a terminal/console program, students are advised to become familiar with this interface and related programmes. Tutorials on basic aspects will be provided. Any text editor (emacs, notepad) can be used.

Learning Resources

The following online learning resources are available for this course, at <http://www.may.ie/academic/music/musictec>

- (a) **Lecture slides.**
- (b) **Background texts and papers.**
- (c) **Csound code examples** for each lecture topic.
- (d) **Software manuals, guides and other reference material.**
- (e) **Links to relevant sites**

Recommended Reading

The following books are recommended as the basic course literature:

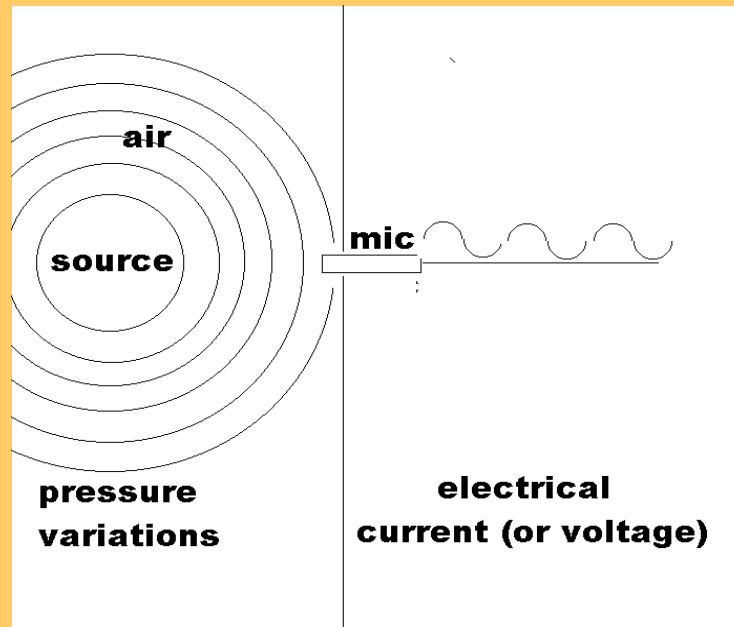
Boulanger, R (Ed.) (2000). *The Csound Book*. MIT Press, Cambridge, Mass.

Dodge, C & Jerse, T (1998). *Computer Music*. Schirmer Books, New York.

These books are available at the library, but students are highly advised to obtain their own copies.

Analog signals

When we capture a sound using a microphone, we are creating an *analog* version of the mechanical pressure variations in the air, in the form of varying electrical current (or voltage).

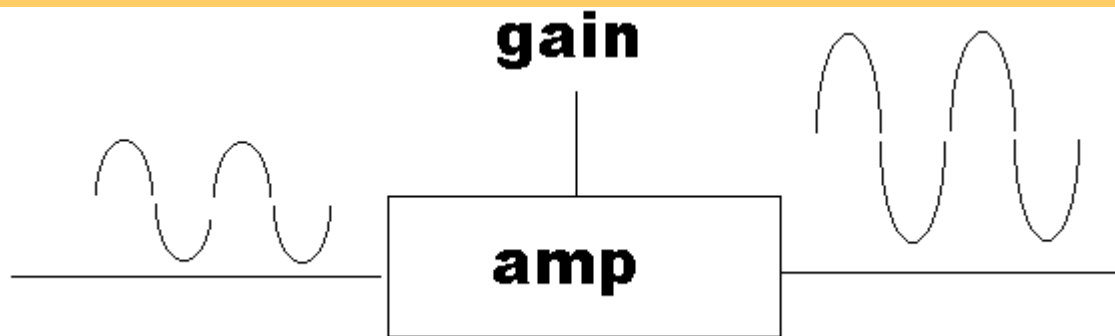


The *continuous* variation of pressure is *transduced* into *continuous* variation of voltage or current.

Microphones are transducers

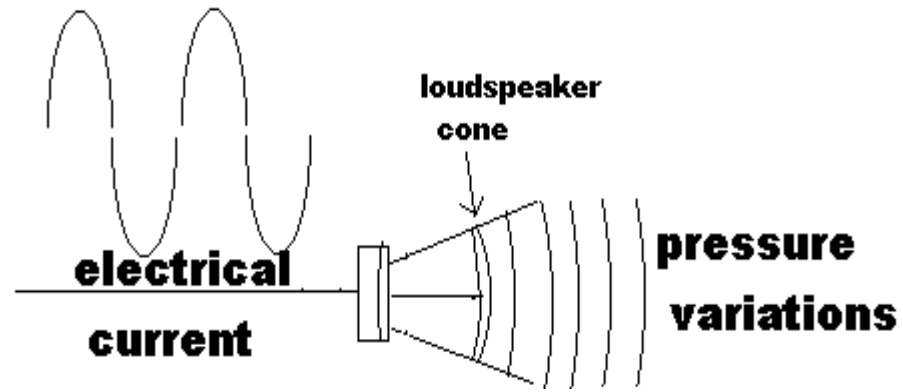
They transform one type of signal into another, without altering its *nature*. The accuracy with which these transducers convert the information strengthens the ties linking electrical signals and their mechanic counterparts.

Analog signals can be transmitted and further modified, amplified or attenuated. An *amplifier* multiplies the analog signal by a constant called the *gain* of the amplifier.



Loudspeakers are also transducers

When we play back a tape through loudspeakers, we are doing the *opposite* transduction process. The loudspeaker vibrates with the electric signal and transforms it into air pressure variations



Again, the acoustic and electric signals are analog versions of each other. The loudspeaker is the *inverse transducer* to the *microphone*. It reproduces the original sound field at the point measured by the microphone.

Transduction

Ideally, the transduction process should be transparent: what comes in comes out unspoilt. If that was the case, then

voltage variation \propto air-pressure variation

(\propto means 'is proportional to')

But in real-world situations, this is never the case. Noise and distortion are always inserted into the signal. Every time it passes through transduction or it is transmitted electrically, resulting in loss of signal. So, a more realistic picture shows

voltage variation \propto distortion[air-pressure variation] + noise

If distortion is null, then the system is said to be linear.

RMS amplitude

When measuring the varying amplitude of a signal that rises and falls above a mean value (generally 0), we use an averaging method called the ‘root-mean-square’ (rms) method:

1. Amplitude values are taken at regular spaced times.
2. These values are squared and averaged
4. The RMS amplitude is the square root of that result.

RMS amplitudes are proportional to *peak amplitudes* (the max absolute value of a signal), but their actual value will depend on the *type* of signal being measured. *Different waveshapes* with the same peak amp will yield *different RMS amplitudes*.

Noise

Noise is defined here as any *unwanted signal* added to the system. The loss of signal caused by noise can be determined by a simple relationship: the **signal-to-noise (SNR)** ratio.

It measures the ratio in levels of the original signal and the noise floor (the inherent noise in a system).

$$\text{SNR} = \frac{\text{max signal (rms) amp}}{\text{max noise (rms) amp}}$$

(The rms amplitude is the root-mean-squared average amplitude of a time-varying signal)

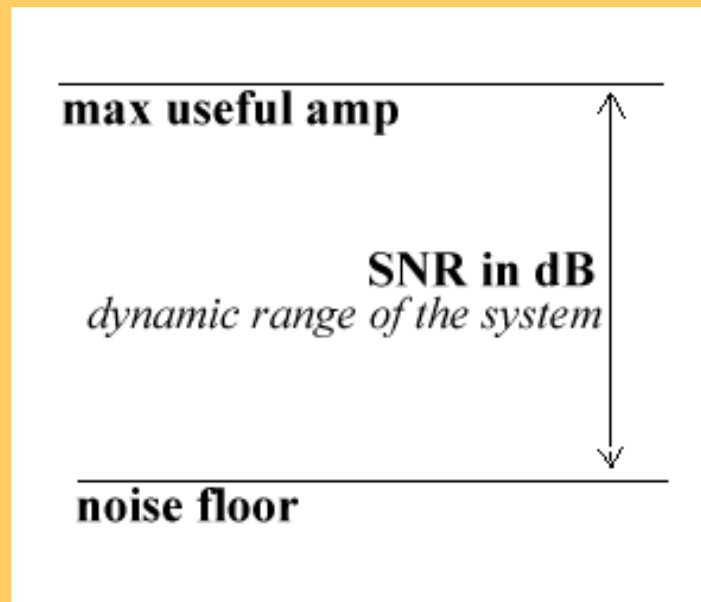
It is common to express the SNR ratio in the *dB scale*:

$$\text{SNR} = 20 \log \frac{\text{max signal (rms) amp}}{\text{max noise (rms) amp}}$$

Dynamic Range

The most important result of a poor signal-to-noise ratio is a reduced dynamic range: if players play too softly, their sound is going to be masked by noise.

Professional analog systems can have very good SNRs (above 90 dB), but home equipment can suffer a lot from poor SNRs.



Distortion

Distortion is what limits the maximum useful amplitude, as it normally increases as the signal increases. So the max amplitude will be the one at which distortion is kept below some tolerance level.

Normally distortion can be divided into three types:

frequency distortion, caused by a device's inability to respond the same way to all frequencies;

amplitude distortion, caused by non-linearities in the response of a device to various values of input amplitudes;

phase distortion, caused by the difference in *time response* to different frequencies, introducing delays at various ranges of complex input signal.

Analog Systems Problems

In general, analog systems can be very unreliable when it comes to noise and distortion. Also, every time something is copied or transmitted an amount of noise/distortion is introduced in the process. If this is done many times, the cumulative effect can deteriorate the signal quite a lot.

Digital technology offers a solution. In digital systems, sound is stored as numbers, so a signal can be effectively “cloned”. Also, mathematical routines can be applied to prevent errors in transmission, which could introduce noise in the signal.

Digital Signals

Digital signals can be seen as a form of *encoding* of analog signals. The resulting signal is not the same as the original, because of the encoding process. The recuperation of the original signal requires a *decoding* process.

These processes are supposed to be complementary, the decoded signal must (within limits) match the original signal.

The most common form of **analog-to-digital conversion** is **PCM** (Pulse Code Modulation). This transforms the original analog signal into a stream of **binary digits** (BITS), **0s** and **1s**.

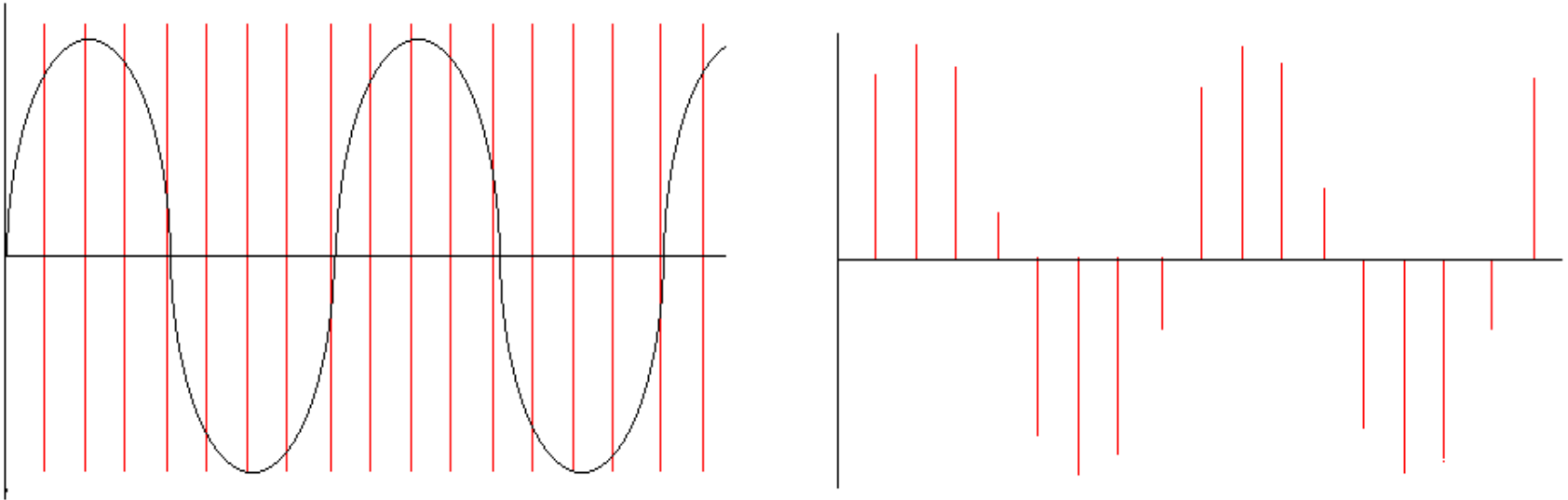
Linear PCM Digital Audio Encoding

Analog to Digital Conversion (ADC)

1. Filter the signal with low-pass filter, to cut high-frequency signals. Cut-off frequency around (15 - 20% lower than) **1/2 sampling rate**
2. Sample at **sampling rate** times/sec
3. Quantise at 2^N levels (N is the resolution in bits)

Before we discuss step no.1, let's have a look at what **sampling** is!

Sampling:



At every sampling period, a value is obtained from the input analog signal. This value is called a *sample*. The sampling period is defined by a sampling frequency or *sampling rate* (SR), as $1/\text{SR}$, or its inverse. The usual sampling rate (for CD-quality audio) is 44100Hz or 44.1KHz (or *samples per sec*).

THE SAMPLING THEOREM

To represent digitally a signal containing frequency components up to **X Hz**, it is necessary to use a sampling rate of at least **2X samples per second**

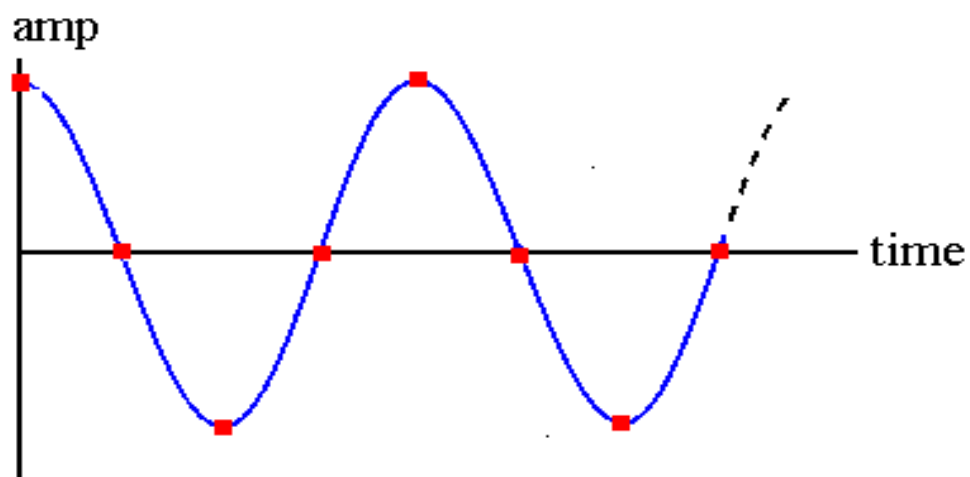
According to this theorem, the system will not be able to represent any frequency above $1/2$ sampling rate (aka the *Nyquist* frequency).

Anything above that will be misrepresented by a frequency below the Nyquist !

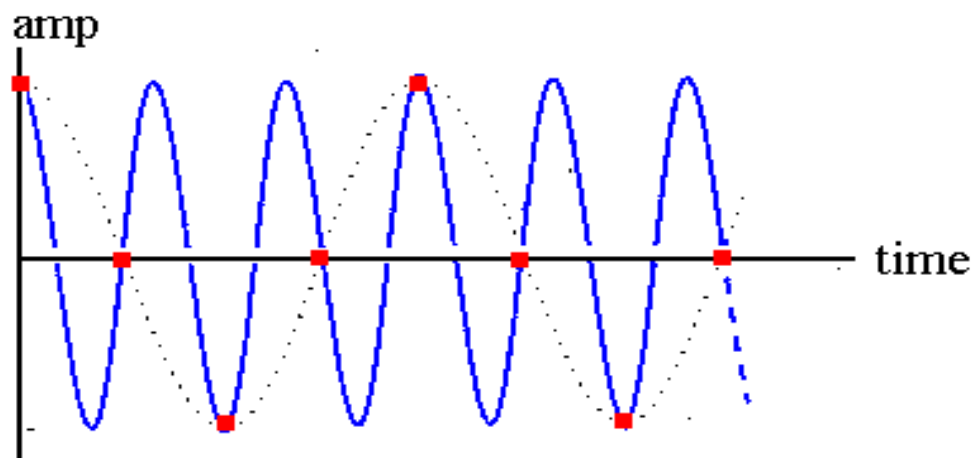
Aliasing

SR = 40,000 samples/sec

10 KHz
sine wave



30 KHz
sine wave
dotted lines
show a
10 KHz
sine wave



Encoded Frequencies

The relation between the **represented (output)** frequency and the **sampled (input)** frequency is shown by the equation below:

$$f_{out} = f_{in} - \left[\text{int} \left(\frac{2f_{in}}{SR} \right) \right] SR$$

f_{out} , the represented frequency

f_{in} , the sampled frequency

SR , the sampling rate

Examples, for SR=40000 Hz

$$f_{out} = f_{in} - \left[\text{int} \left(\frac{2f_{in}}{SR} \right) \right] SR$$

(a) 10000 Hz

$$f_{out} = 10000 - \left[\text{int} \left(\frac{2 \times 10000}{40000} \right) \right] 40000 = 10000 \text{ Hz}$$

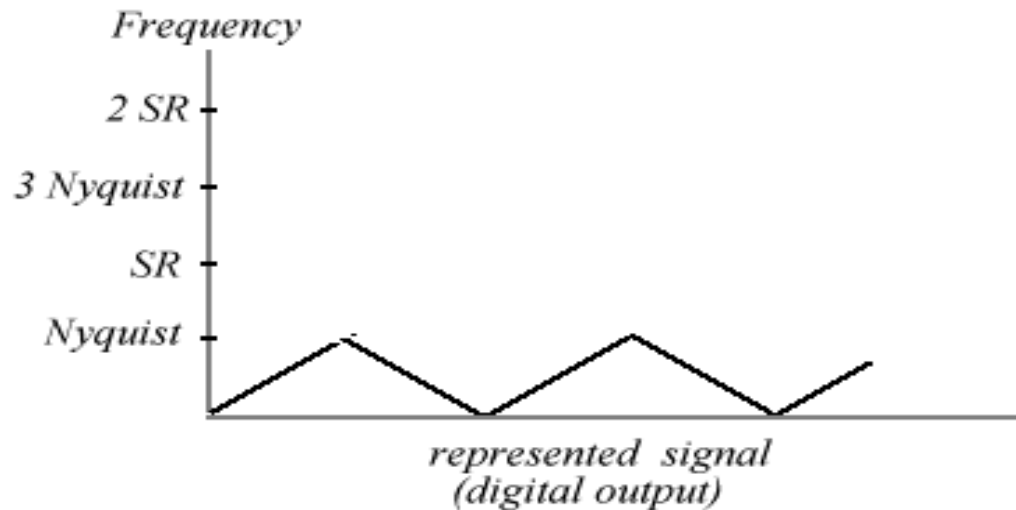
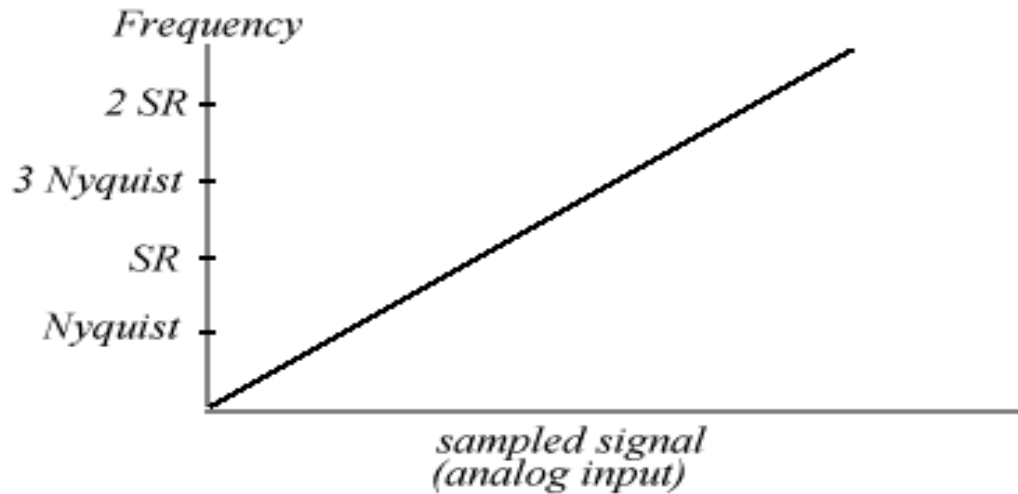
(b) 20000 Hz

$$f_{out} = 20000 - \left[\text{int} \left(\frac{2 \times 20000}{40000} \right) \right] 40000 = -20000 \text{ Hz}$$

(c) 30000 Hz

$$f_{out} = 30000 - \left[\text{int} \left(\frac{2 \times 30000}{40000} \right) \right] 40000 = -10000 \text{ Hz}$$

Comparison of Input and Output Frequencies



Low-pass filtering

Why?

When you sample a signal at K times/sec, the highest frequency which is possible to represent is $K/2$ cycles/sec.

Everything above this must be eliminated !

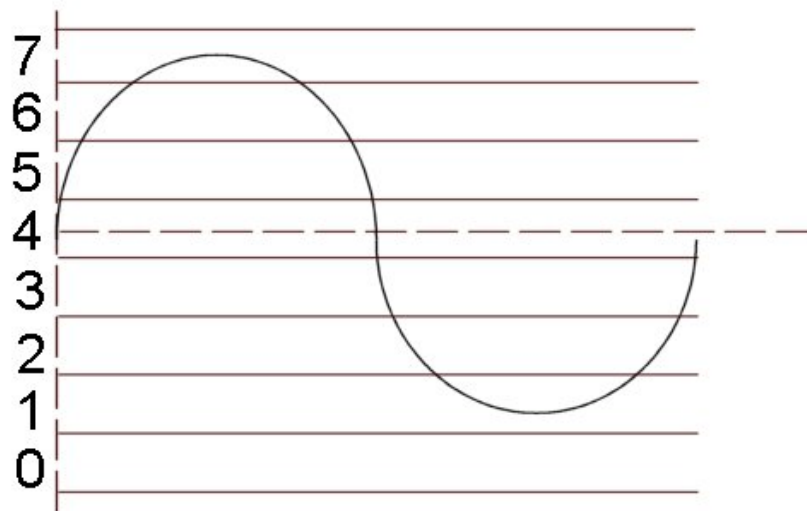
All frequencies above that will be misrepresented, that is, they will come out as some other frequency. This is called *aliasing* or *foldover*. Anything above $K/2$ Hz will be *mirrored* around $K/2$ Hz, for example:

if $K = 20000\text{Hz}$ ($K/2 = 10\text{ KHz}$), an input frequency of **13000Hz** will be misrepresented as **7000Hz**, and a frequency of **15000Hz** will come out as **5000Hz**.

Quantisation

Every sample obtained is set to a specific quantisation level. The number of levels depends on the precision of the measurement, in bits. This means how many binary digits are used to store the samples. The usual, CD-quality, standard is 16-bit, giving **65434** levels (2^{16}).

The example below shows a 3-bit (8-level) quantisation in relation to an input signal:



Linear Quantisation

The quantiser assigns a certain sampled value a certain region, and the number of regions depends on the *range of numbers* available to represent a value in the system.

For instance, if we are representing a number using *1 bit* only, there are only *two* distinct values it can assume (two regions of quantisation), namely *0 or 1*.

If we add another bit, then we can have more states (or values or regions), **four** in total: **00, 01, 10, 11**.

In general, for *n* bits, we can have 2^n states.

In the case of *linear* quantisation, the quantiser divides the available range of voltages into 2^n regions, each of them of the *same size*.

Quantisation Error and SNR

Any voltage between a *minimum* and a *maximum* of a particular region will be mapped into a *single value*. This value is taken to be the one at the mid-point between the max and the min.

If the value sampled is anywhere else but mid-point, there will be an error, and the max possible error is *half the size* of the region.

The nature of this error is random and constant, so it can be taken as a form of *noise* added to the signal. A **SNR** measurement can be estimated to give us an idea of the dynamic range of the system.

Signal-to-quantisation-error-noise-ratio (SQNR)

The *SQNR* for a system will, of course, depend on the size of the quantisation regions, which in turn depend on their number. In general, for each bit we gain 6 dB of SQNR. So a 16-bit system will have a maximum SQNR of 96 dB.

In addition to this, the SQNR will also depend on the strength of the signal, a softer signal will be more affected to noise than a louder. The SQNR is:

$$\text{SQNR (dB)} = 6N + S$$

where N is the number of bits used and S is the signal amplitude in relation to full scale (0dB)

Digital Signal Decoding

In order for us to listen to a digital recording, the sound has to be converted back into analog form, before it is transduced into mechanical waves.

This is done by the DAC, digital-to-analog converter. It effectively reverses the ADC process, as it:

1. Finds the voltages for each quantisation level
2. At every sampling period, outputs a voltage
3. Low-pass filters the signal, to smooth it
(smoothing out the step-like shape of the output wave)

Oversampling and other methods

The above method is the basic DAC conversion. Because it relies on expensive analog hardware for the filtering, in certain cases it is not adopted.

Other methods, such as oversampling with *sigma-delta* modulation are common solutions for the problem. They are based on changing the sampling frequency to a higher value, so that the analog filtering can be done with simpler (and cheaper) hardware.

This is the basic way the converters in most CD players work. Some of them use what is known as *1-bit* conversion. In any case any other methods of PCM signal conversion are effectively equivalent to the ones described here.