Chapter 4
Digital Audio Representation

CS 3570

Objectives

- Be able to apply the Nyquist theorem to understand digital audio aliasing.
- Understand how dithering and noise shaping are done.
- Understand the algorithm and mathematics for μ-law encoding.
- Understand the application and implementation of the Fourier transform for digital audio processing.
- Understand what MIDI is and the difference between MIDI and digital audio wave.

Introduction

- Sound is a mechanical wave that is an oscillation of pressure transmitted through a solid, liquid, or gas.
- The perception of sound in any organism is limited to a certain range of frequencies(20Hz~20000Hz for humans).
- How do we process “sound”?
  - The changing air pressure caused by sound is translated into changing voltages.
  - The fluctuating pressure can be modeled as continuously changing numbers—a function where time is the input variable and amplitude (of air pressure or voltage) is the output.

Pulse Code Modulation

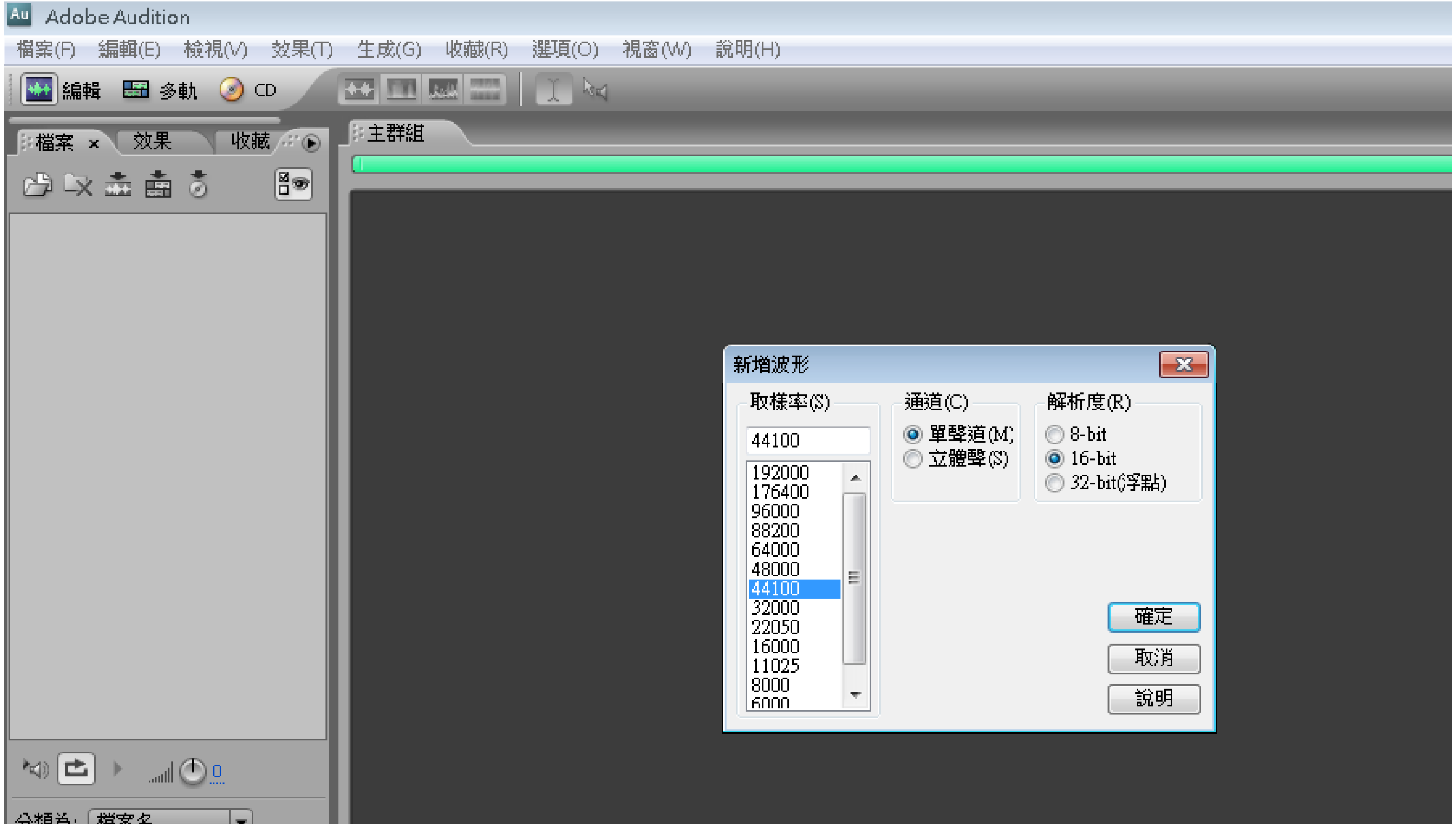
- Pulse-code modulation (PCM) is a method used to digitally represent sampled analog signals.
- A PCM stream is a digital representation of an analog signal, in which the magnitude of the analogue signal is sampled regularly at uniform intervals, with each sample being quantized to the nearest value within a range of digital steps.
- PCM files are digitized but not compressed.
- DPCM (Differential Pulse Code Modulation)

Audio Digitization

- When you create a new audio file in a digital audio processing program, you are asked to choose
  - **Sampling rate**: The sampling rate, sample rate, or sampling frequency defines the number of samples per unit of time (usually seconds) taken from a continuous signal to make a discrete signal.
  - **Bit depth**: Bit depth describes the number of bits of information recorded for each sample.
- For CD quality, the sampling rate is 44.1kHz and the bit depth is 16.

Audio Digitization

- Create a new audio file (Adobe Audition)



新增波形

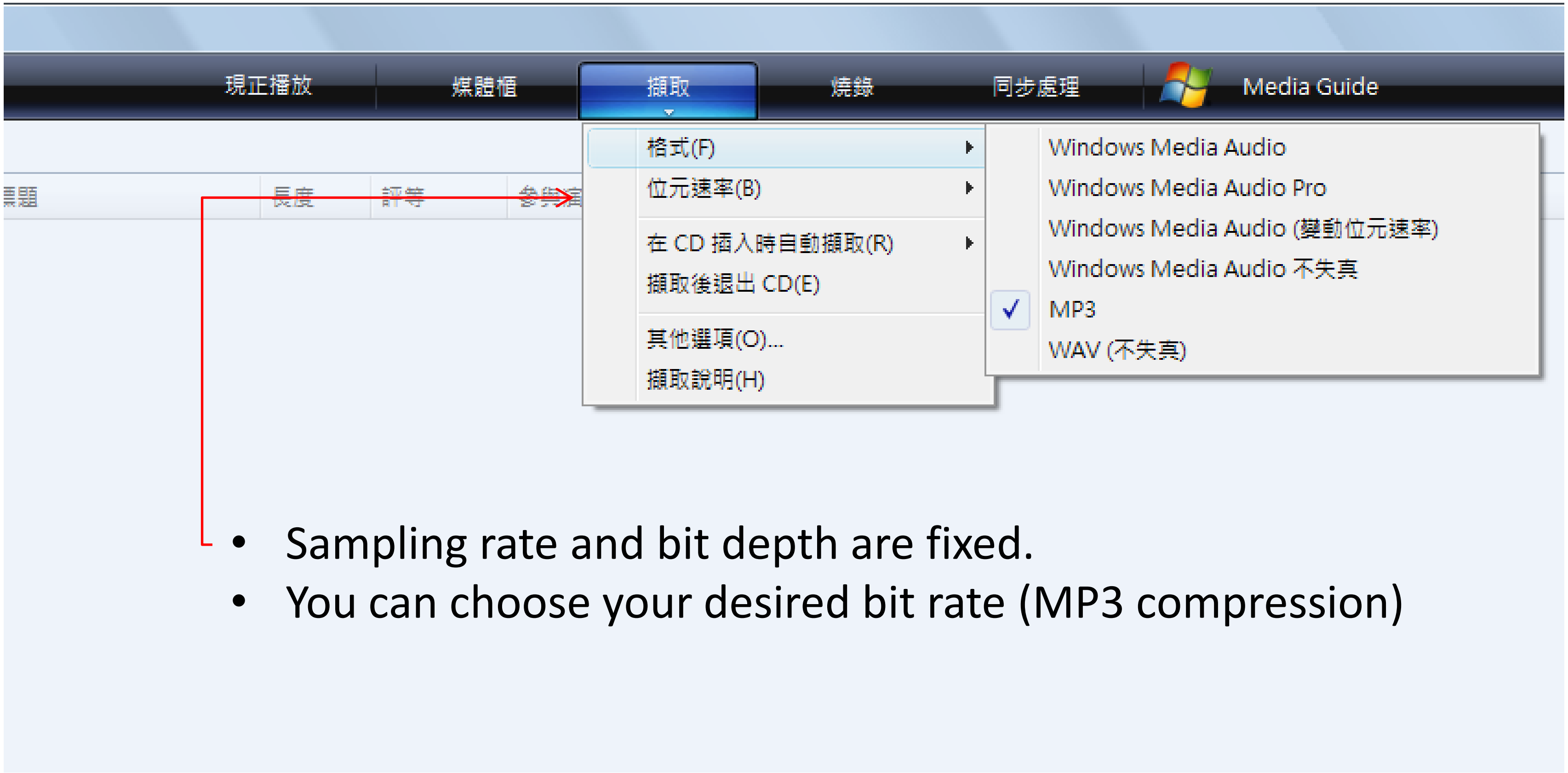
取樣率(S)
44100

通道(C)
● 單聲道(M)
○ 立體聲(S)

解析度(R)
○ 8-bit
● 16-bit
○ 32-bit(浮點)

Audio Digitization

- Extract music from CDs



- Sampling rate and bit depth are fixed.
- You can choose your desired bit rate (MP3 compression)

Nyquist Theorem

- Review
  - Let *f* be the frequency of a sine wave. Let *r* be the minimum sampling rate that can be used in the digitization process such that the resulting digitized wave is not aliased. Then *r=2f.*

Nyquist Theorem

- Nyquist frequency
  - Given a sampling rate, the ***Nyquist frequency*** is the highest actual frequency component that can be sampled without aliasing.
  - Ex:

    If we choose a sample rate of 8000Hz, the Nyquist frequency $f_{nf} = \frac{1}{2} f_{samp} = 4000$Hz

Nyquist Theorem

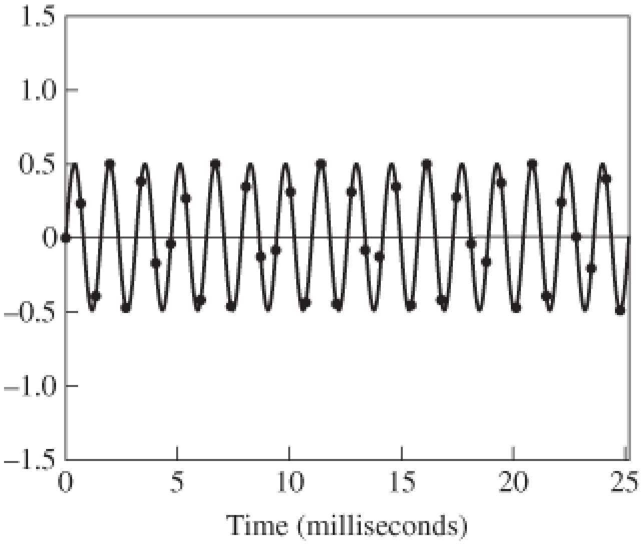
- Nyquist rate
  - Given an actual frequency to be sampled, the ***Nyquist rate*** is the lowest sampling rate that will permit accurate reconstruction of an analog digital signal.
  - Ex:

    If the highest frequency component is 10,000Hz,

    the Nyquist rate $f_{nr} = 2f_{max} =$ 20,000Hz

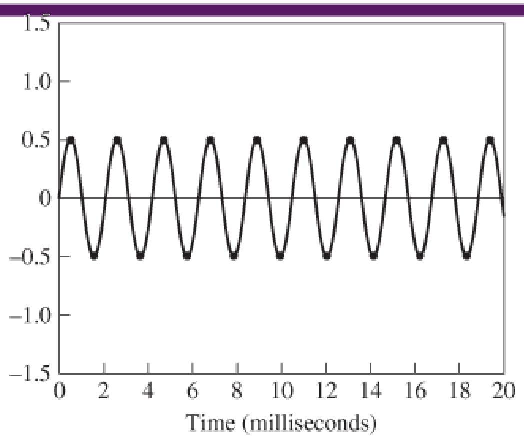
Sampling Rate and Aliasing

- In essence, the reason a too-low sampling rate results in aliasing is that there aren't enough sample points from which to accurately interpolate the sinusoidal form of the original wave.

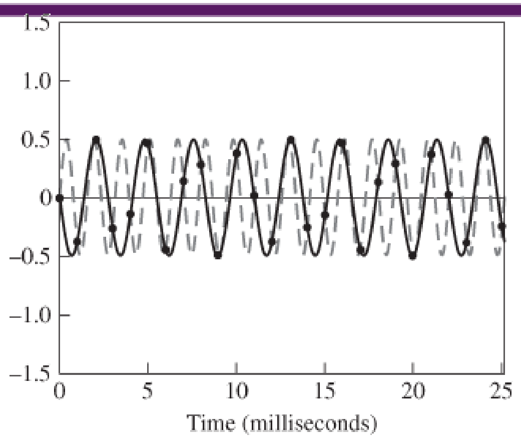


Samples taken more than twice per cycle will provide sufficient information to reproduced the wave with no aliasing

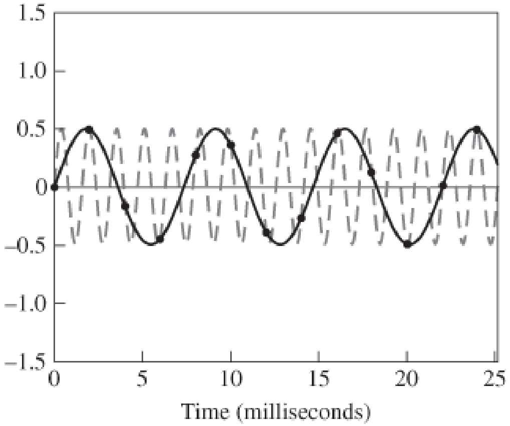
Sampling Rate and Aliasing



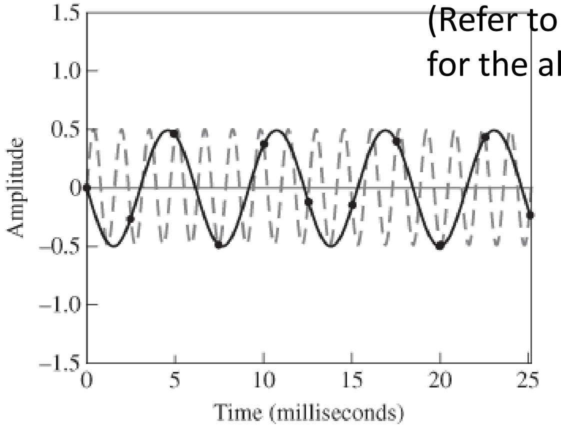
Samples taken exactly twice per cycle *can* be sufficient for digitizing the original with no aliasing



A 637 Hz wave sampled at 1000 Hz aliases to 363 Hz



A 637 Hz wave sampled at 500 Hz aliases to 137 Hz



(Refer to Algo.4.1
for the aliased frequency)

A 637 Hz wave sampled at 400 Hz aliases to 163 Hz

Decibels

- Decibels(*E*0, *I*0: threshold of human hearing)
  - Decibels-sound-pressure-level (dB_SPL)

$$dB\_SPL = 20\log_{10}\left(\frac{E}{E_0}\right), E_0 = 2 \times 10^{-5} Pa$$

  - Decibels-sound-intensity-level (dB_SIL)

$$dB\_SIL = 10\log_{10}\left(\frac{I}{I_0}\right), I_0 = 10^{-12} W/m^2$$

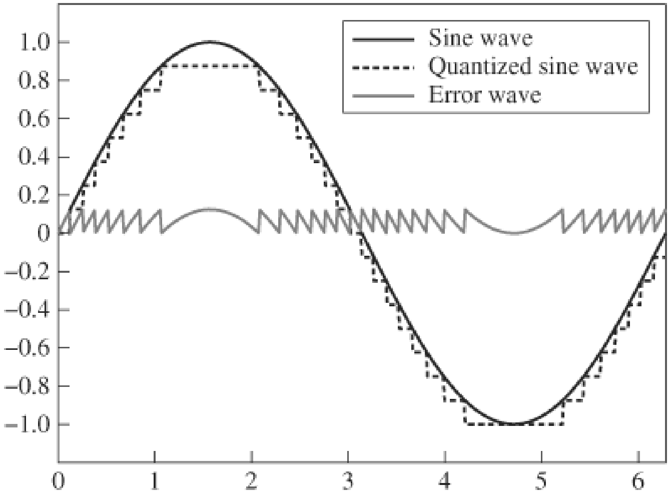
- Decibels can be used to measure many things in physics, optics, electronics, and signal processing.
- A decibel is not an absolute unit of measurement.

Dynamic Range

- Dynamic range is the ratio between the smallest nonzero value, which is 1, and the largest, which is 2^n. The *dynamic range of the audio file, d,* in decibels, is defined as
  $$d = 20\log_{10} 2^n = 20n\, log_{10} 2 \approx 6n$$
- The definition is identical to the definition of SQNR, and this is why you see the terms SQNR and dynamic range sometimes used interchangeably.
- Be careful not to interpret this to mean that a 16-bit file allows louder amplitudes than an 8-bit file. Rather, dynamic range gives you a measure of the range of amplitudes that can be captured relative to the loss of fidelity compared to the original sound.

Audio Dithering

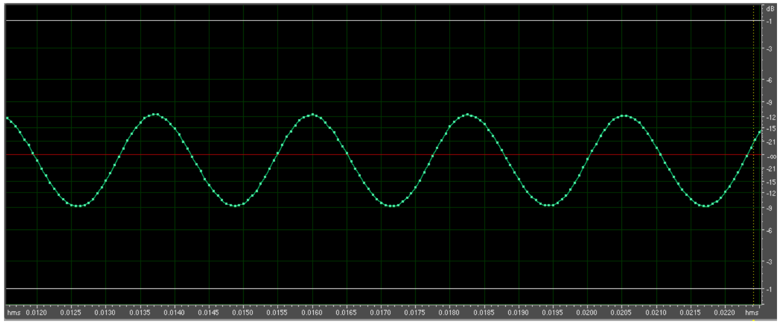
- ***Audio dithering*** is a way to compensate for quantization error.
- Quantized signals would sound 'granular' because of the stair-step effect. The quantized signals sound like the original signals plus the noise.
- The noise follows the same pattern as the original wave, human ear mistakes it as the original signal.



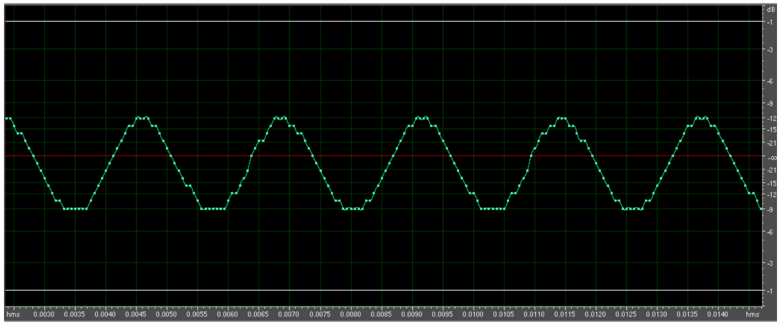
Audio Dithering

- Example1-simpe wave

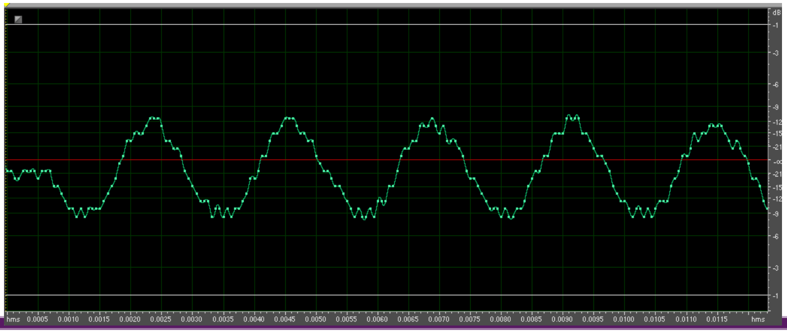
Original wave



After bit reduction

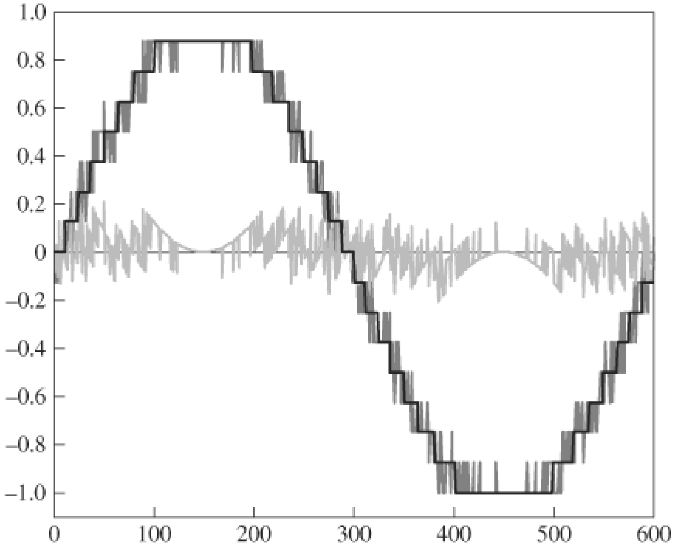


After dithering



Audio Dithering

- Adding a random noise(dither) to the original wave eliminates the sharp stair-step effect in the quantized signal.
- The noise is still there, but has less effect on the original signal.(we can hear the smooth signal without stair-step effect)

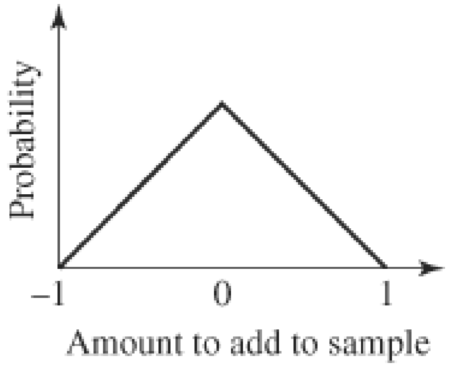


dithered
quantized wave

Audio Dithering

- Dithering function
  - Triangular probability density function (TPDF)

Triangular probability function



  - Rectangular probability density function (RPDF): All numbers in the selected range have the same probability
  - Gaussian PDF: The Gaussian PDF weights the probabilities according to a Gaussian
  - Colored dithering: Colored dithering produces noise that is not random and is primarily in higher frequencies.

Audio Dithering

- Example2 – complex wave

Original wave 🔊

After bit reduction 🔊

After dithering 🔊

Noise Shaping

- ***Noise shaping*** is another way to compensate for the quantization error. Noise shaping is *not* dithering, but it is often used along with dithering.
- The idea behind noise shaping is to redistribute the quantization error so that the noise is concentrated in the higher frequencies, where human hearing is less sensitive, or we can use a low-pass filter to filter out the high frequency components.

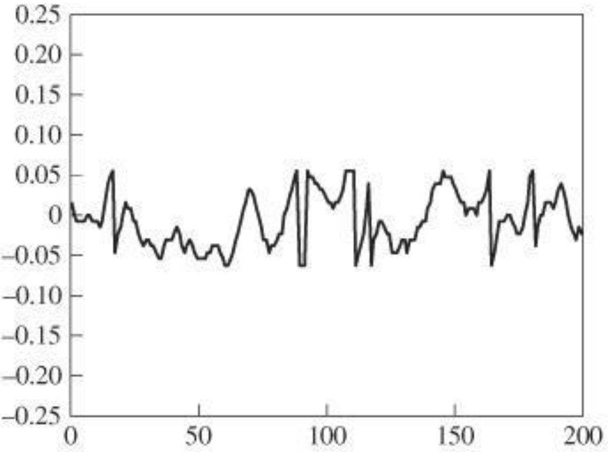
Noise Shaping

- First-order feedback loop for noise shaping
  - Let ***F_in*** be an array of *N* digital audio samples that are to be quantized, dithered, and noise shaped, yielding ***F_out***. For $0 \le i \le N - 1$, define the following: $\boldsymbol{F\_in}_i$ is the *i*th sample value, not yet quantized.
  - $\boldsymbol{D}_i$ is a random dithering value added to the *i*th sample.
  - The assignment statement $\boldsymbol{F\_in}_i = \boldsymbol{F\_in}_i + \boldsymbol{D}_i + \boldsymbol{cE}_{i-1}$ dithers and noise shapes the sample. Subsequently, $\boldsymbol{F\_out}_i = [\boldsymbol{F\_in}_i]$ quantizes the sample.
  - $\boldsymbol{E}_i$ is the error resulting from quantizing the *i*th sample after dithering and noise shaping.
  - For $i = -1$, $\boldsymbol{E}_i = 0$. Otherwise, $\boldsymbol{E}_i = \boldsymbol{F\_in}_i - \boldsymbol{F\_out}_i$.

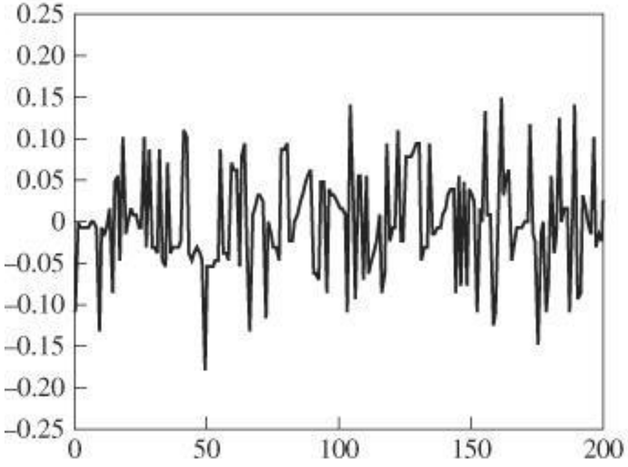
Noise Shaping

- What does noise shaping do?
  - Move noise's frequency to above the Nyquist frequency, and filter it out. We are not losing anything we care about in the sound.
- The term *shaping* is used because you can manipulate the "shape" of the noise by manipulating the noise shaping equations
- The general statement for an *n*th order noise shaper noise shaping equation
  becomes $\boldsymbol{F\_out}_i = \boldsymbol{F\_in}_i + \boldsymbol{D}_i + c_{i-1}\boldsymbol{E}_{i-1} + c_{i-2}\boldsymbol{E}_{i-2} + \cdots$
  $+ c_{i-\mathrm{n}}\boldsymbol{E}_{i-\mathrm{n}}$.

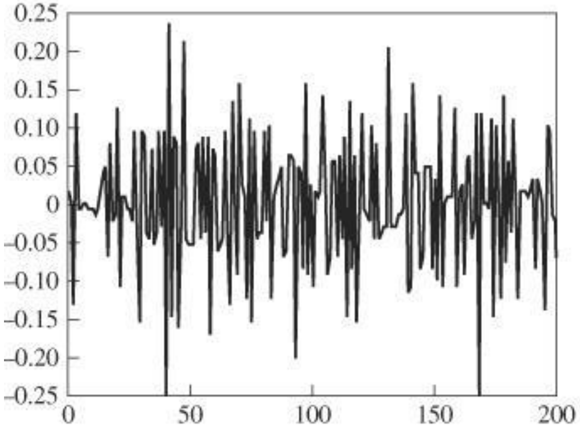
Noise Shaping



Quantization error wave with no dithering



Quantization error wave with dithering



Quantization error wave with dithering and noise shaping

Noise Shaping

- Original 🔊
- After bit reduction 🔊
- After dithering 🔊
- Dithering with noise shaping 🔊

Non-Linear Quantization

- ***Nonlinear encoding***, or ***companding,*** is an encoding method that arose from the need for compression of telephone signals across low bandwidth lines. Companding means compression and then expansion.
- How this works?
  - Take a digital signal with bit depth *n* and requantize it in *m* bits, *m* < *n*, using a nonlinear quantization method.
  - Transmit the signal.
  - Expand the signal to *n* bits at the receiving end.
- Why not just use linear quantization?

Non-Linear Quantization

- Reasons for non-linear quantization
  - Human auditory system is perceptually non-uniform. Humans can perceive small differences between quiet sounds, but not for louder sounds.
  - Quantization error generally has more impact on low amplitudes than on high ones, why?

    0.499 -> 0, err=(0.499-0)/0.499 = 100%

    126.499 -> 126, err=(126.499-126)/126.499 = 0.4%
- Use *more* quantization levels for low amplitude signals and *fewer* quantization levels for high amplitudes.

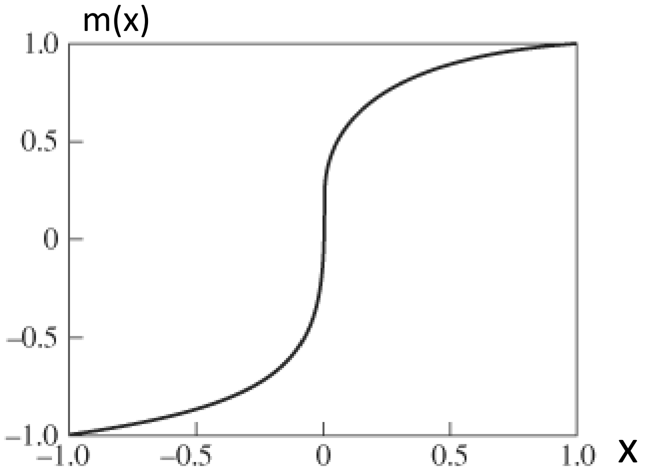
μ-law Function

- Let *x* be a sample value normalized so that −1 ≤ *x* < 1. Let *sign*(*x*) = −1 if *x* is negative and *sign*(*x*) = 1 otherwise. Then the ***μ-law function*** is defined by

$$m(x) = sign(x)(\frac{\ln(1+\mu|x|)}{\ln(1+\mu)})$$

$$= sign(x)\left(\frac{\ln(1+255|x|)}{5.5452}\right), for\ \mu = 255$$

- The μ-law function has a logarithmic shape. Its effect is to provide finer-grained quantization levels at low amplitudes compared to high.



μ-law Function

- μ means the new quantization level, 255(8 bits) in the North American and Japanese standards.
- Then the ***inverse μ-law function*** is defined by

$$
\begin{aligned}
d(x) &= sign(x)\left(\frac{(\mu+1)^{|x|}-1}{\mu}\right) \\
&= sign(x)\left(\frac{256^{|x|}-1}{255}\right), for\ \mu = 255
\end{aligned}
$$

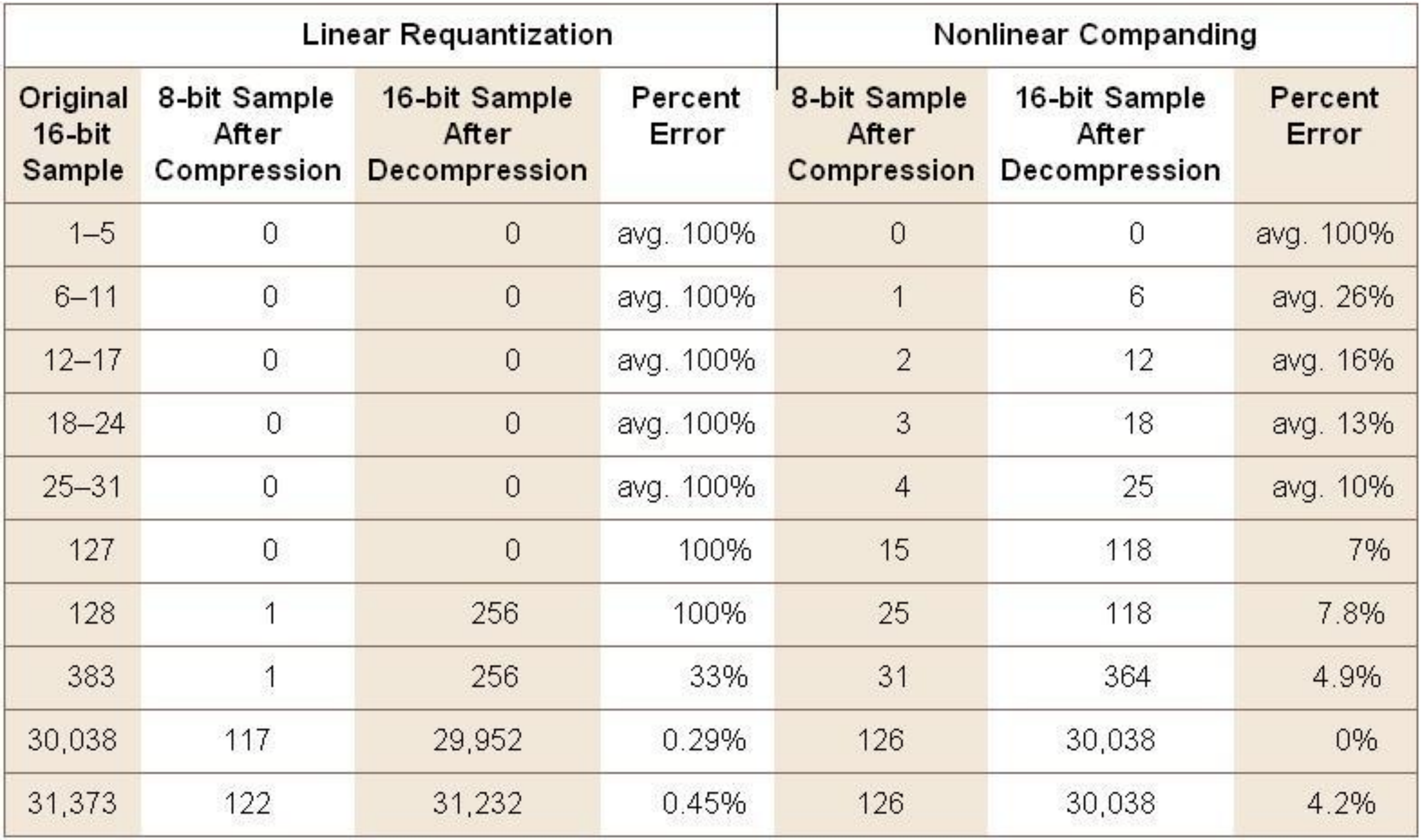
- Let’s see some examples

μ-law Function

- Assume the original signal is quantized with bit depth of 16.

Sample value	Normalized value	m(x)	Scale to 8-bit value	d(x)	Scale back to 16-bit value
16	$\left(\frac{16}{32768}\right) = 0.000488$	≈0.02	$\lfloor 0.02 * 128 \rfloor = 2$	$d\left(\frac{2}{128}\right) = 0.00035$	$\lfloor 0.00035 * 32768 \rfloor = 11$
30037	$\left(\frac{30037}{32768}\right) = 0.9167$	≈0.9844	$\lfloor 0.9844 \times 128 \rfloor = 125$	$d\left(\frac{125}{128}\right) = 0.8776$	$\lfloor 0.8776 * 32768 \rfloor = 28758$

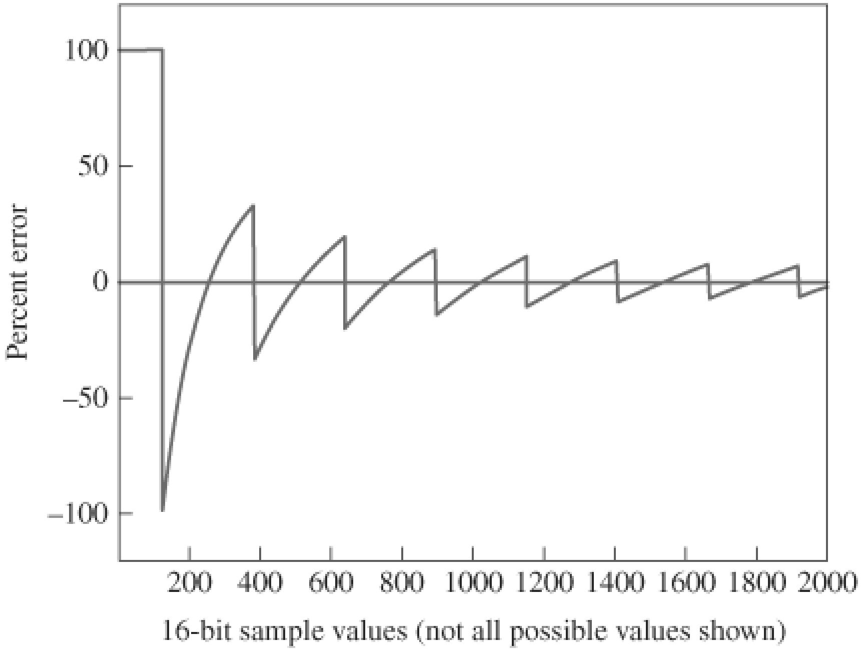
Linear vs. Non-Linear Requantization



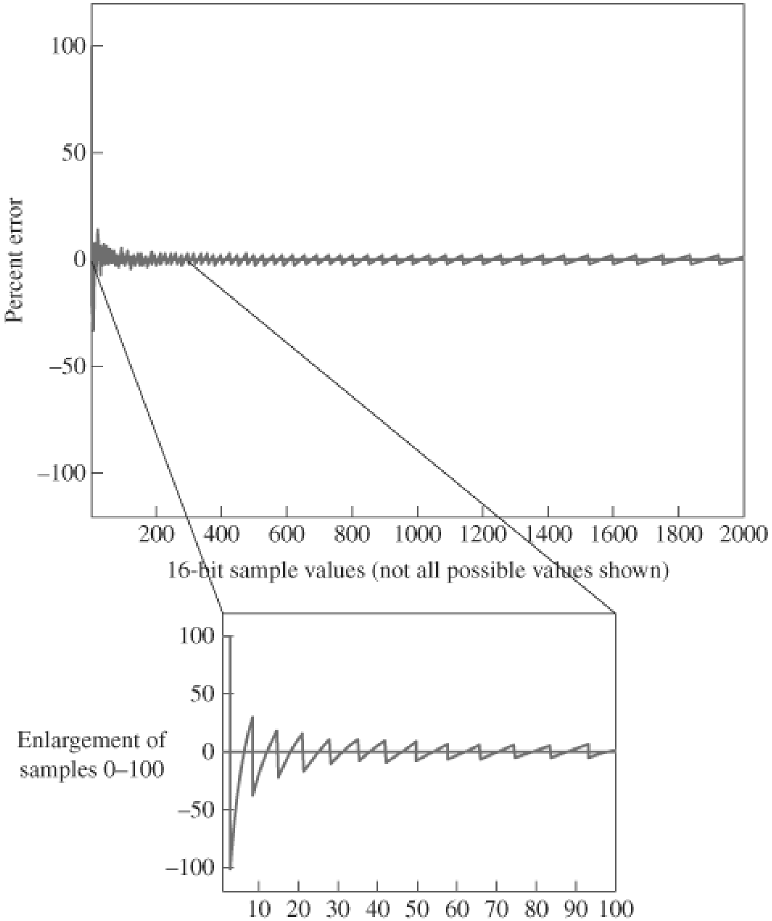
	Linear Requantization			Nonlinear Companding		
Original 16-bit Sample	8-bit Sample After Compression	16-bit Sample After Decompression	Percent Error	8-bit Sample After Compression	16-bit Sample After Decompression	Percent Error
1–5	0	0	avg. 100%	0	0	avg. 100%
6–11	0	0	avg. 100%	1	6	avg. 26%
12–17	0	0	avg. 100%	2	12	avg. 16%
18–24	0	0	avg. 100%	3	18	avg. 13%
25–31	0	0	avg. 100%	4	25	avg. 10%
127	0	0	100%	15	118	7%
128	1	256	100%	25	118	7.8%
383	1	256	33%	31	364	4.9%
30,038	117	29,952	0.29%	126	30,038	0%
31,373	122	31,232	0.45%	126	30,038	4.2%

Linear vs. Non-Linear Requantization

- Error of linear requantization

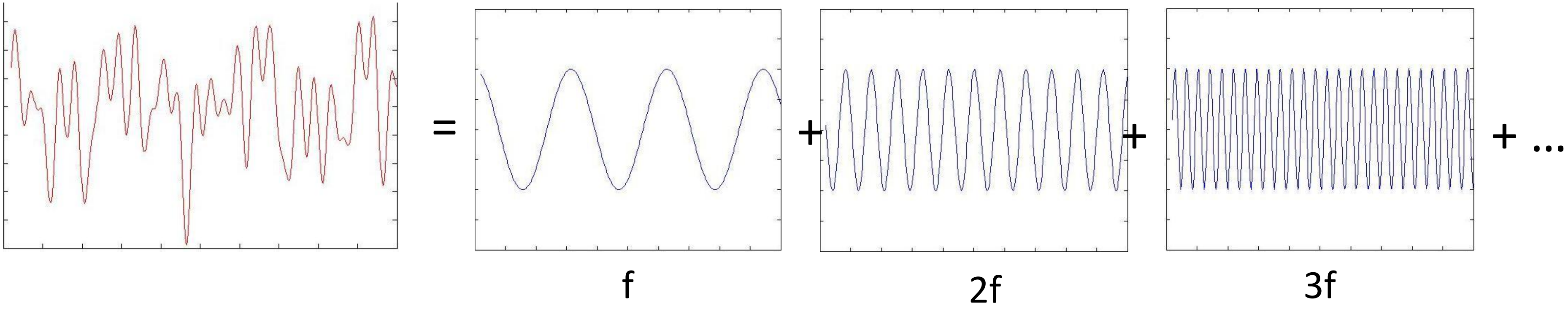


- Error of non-Linear requantization



Frequency Analysis

- Time domain
  - Input: time (x-axis)
  - Output: amplitude(y-axis)
- A complex waveform is equal to an infinite sum of simple sinusoidal waves, beginning with a ***fundamental frequency*** and going through frequencies that are integer multiples of the fundamental frequency – ***harmonic frequencies***.

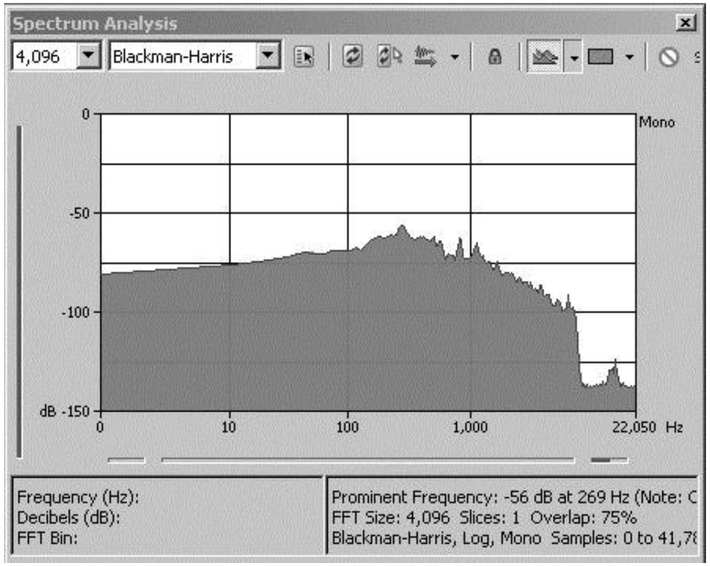


Frequency Analysis

- Two views for frequency analysis
  - Frequency analysis view(spectrum analysis view)- common

    x-axis: frequency

    y-axis: magnitude of the frequency component



4,096

Blackman-Harris

Frequency (Hz):
Decibels (dB):
FFT Bin:

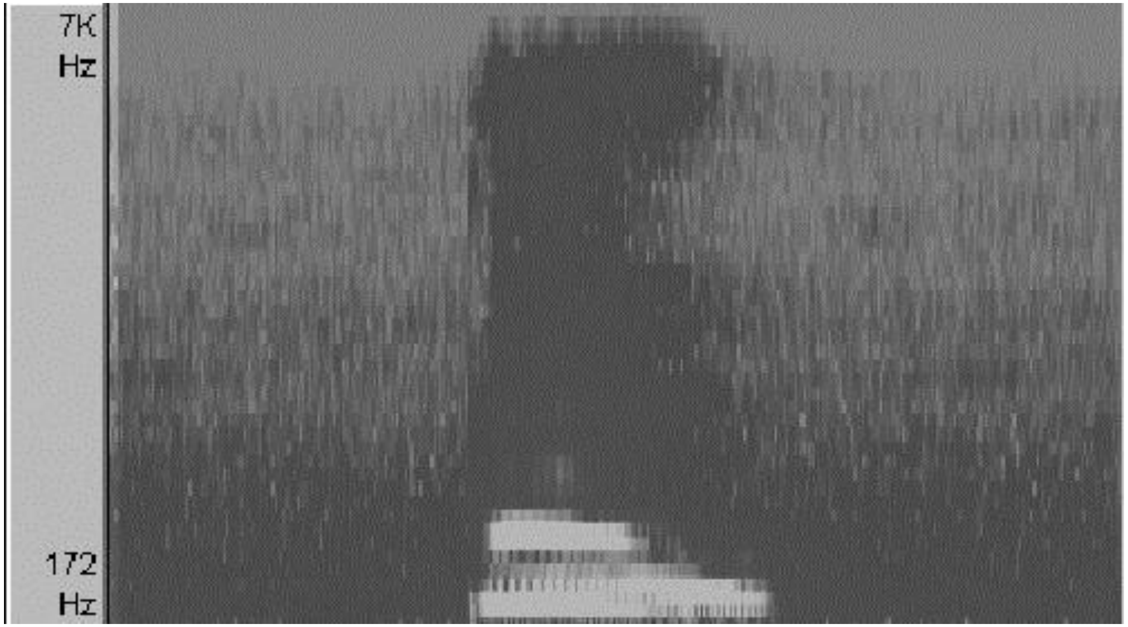
Prominent Frequency: -56 dB at 269 Hz (Note: C
FFT Size: 4,096 Slices: 1 Overlap: 75%
Blackman-Harris, Log, Mono Samples: 0 to 41,7

Frequency Analysis

- Spectral view

  x-axis: time, y-axis: frequency

  color: magnitude of the frequency component



The Fourier Series

- A ***Fourier series*** is a representation of a periodic function as an infinite sum of sinusoidals:

$$f(t) = \sum_{n=-\infty}^{\infty}[\boldsymbol{a}_n \cos(n\omega t) + \boldsymbol{b}_n \sin(n\omega t)] \quad (4.2)$$

- $\omega = 2\pi f$: fundamental angular frequency

  f: fundamental frequency(f , $f(t)$ are different things)

  $\boldsymbol{a}_n$ and $\boldsymbol{b}_n$ tell how much each of these component frequencies contributes to $f(t)$.

The Fourier Series

- Rewrite (4.2) in a different form

$$f(t) = \boldsymbol{a}_0 + \sum_{n=-\infty}^{-1} [\boldsymbol{a}_n \cos(n\omega t) + \boldsymbol{b}_n \sin(n\omega t)] + \sum_{n=1}^{\infty} [\boldsymbol{a}_n \cos(n\omega t) + \boldsymbol{b}_n \sin(n\omega t)] \qquad (4.3)$$

- $\boldsymbol{a}_0$ is the ***DC component,*** which gives the average amplitude value over one period.

$$\boldsymbol{a}_0 = \frac{1}{T}\int_{-T/2}^{T/2} f(t)dt$$

The Fourier Series

- The other terms in (4.3) are called ***AC components.*** The coefficients of each of these frequency components are

$$\boldsymbol{a}_n = \frac{1}{T}\int_{-T/2}^{T/2} f(t)cos(n\omega t)dt,\ \boldsymbol{b}_n = \frac{1}{T}\int_{-T/2}^{T/2} f(t)sin(n\omega t)dt$$

$$for -\infty \leq n \leq \infty$$

- Since cos(-x)=cos(x), sin(-x)=-sin(x), (4.3) becomes

$$f(t) = \boldsymbol{a}_0 + 2\sum_{n=1}^{\infty}[\boldsymbol{a}_n \cos(n\omega t) + \boldsymbol{b}_n \sin(n\omega t)] \qquad (4.4)$$

The Fourier Series

- There's another, equivalent way of expressing the Fourier series.

$f(t) = \sum_{n=-\infty}^{\infty} \boldsymbol{F}_n e^{in\omega t}$ (4.5)

$\boldsymbol{F}_n = \boldsymbol{a}_n - i\boldsymbol{b}_n$

$e^{in\omega t} = \cos(n\omega t) + isin(n\omega t)$ ----> Euler's formula

- Fourier transform is important in signal processing. It decomposes the signal into different frequency components so that we can analyze it, and do some modification on some of them.

Discrete Fourier Transform

- The ***discrete Fourier transform*** (**DFT**) operates on an array of *N* audio samples, returning cosine and sine coefficients that represent the audio data in the frequency domain.

$$F_n = \frac{1}{N}\sum_{k=0}^{N-1} f_k \cos\left(\frac{2\pi nk}{N}\right) - if_k \sin\left(\frac{2\pi nk}{N}\right) \qquad (4.8)$$

$$= \frac{1}{N}\sum_{k=0}^{N-1} f_k e^{\frac{-i2\pi nk}{N}}$$

$$F(u) = \sum_{r=0}^{M-1} \frac{\sqrt{2}C(u)}{\sqrt{M}} f(r)\cos\left(\frac{(2r+1)u\pi}{2M}\right) \text{DCT, Eq(2.2)}$$

Discrete Fourier Transform

- Let $\boldsymbol{f}_k$ be a discrete integer function representing a digitized audio signal in the time domain, and $\boldsymbol{F}_n$ be a discrete, complex number function representing a digital audio signal in the frequency domain. Then the ***inverse discrete Fourier transform*** is defined by

$$\boldsymbol{f}_k = \sum_{n=0}^{N-1}\left[\boldsymbol{a}_n \cos\left(\frac{2\pi nk}{N}\right) + \boldsymbol{b}_n \sin\left(\frac{2\pi nk}{N}\right)\right] \qquad (4.7)$$

$$= \sum_{n=0}^{N-1} F_n e^{\frac{i2\pi nk}{N}}$$

- Subscript k: signal value at time k

  Subscript n: nth frequency component

Discrete Fourier Transform

- The DC component, $\boldsymbol{a}_0$, is defined by $\boldsymbol{a}_0 = \frac{1}{N}\sum_{k=0}^{N-1} \boldsymbol{f}_k$, giving the average amplitude.
- The AC components are, for 1≤n≤N,

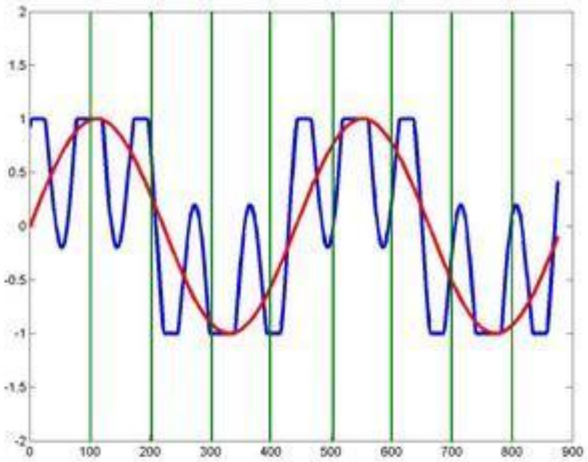
  $$\boldsymbol{a}_n = \frac{1}{N}\sum_{k=0}^{N-1} \boldsymbol{f}_k \cos(\frac{2\pi nk}{N})$$

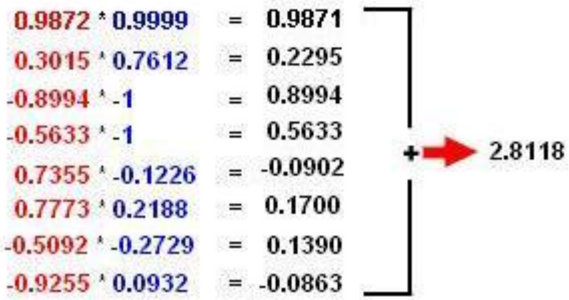
  $$\boldsymbol{b}_n = \frac{1}{N}\sum_{k=0}^{N-1} \boldsymbol{f}_k sin(\frac{2\pi nk}{N})$$

- Fundamental frequency $f = \frac{1}{N}$
- Fundamental angular frequency $\omega = 2\pi f = 2\pi/N$

How does DFT Work?

- Suppose the blue wave represents the complex audio and the red one is a sinusoidal wave of a certain frequency n.
- Green line means the sample points, N=8.
- If the sinusoidal wave fits the signal well, then **F**n is large, which means this frequency component takes a big ratio in the complex signal.





0.9872 * 0.9999 = 0.9871

0.3015 * 0.7612 = 0.2295

-0.8994 * -1 = 0.8994

-0.5633 * -1 = 0.5633

0.7355 * -0.1226 = -0.0902

0.7773 * 0.2188 = 0.1700

-0.5092 * -0.2729 = 0.1390

-0.9255 * 0.0932 = -0.0863

+ → 2.8118

Comparison between DCT and DFT

- You may think now that the DCT is inherently superior to the DFT because it doesn't trouble you with complex numbers, and it yields twice the number of frequency components. But how about the phase information?
- The DFT contains both real part and imaginary part, and thus we can get the phase information. The DCT, however, cancels out the sine terms, together with the phase information.

Phase Information in DFT

- Let the equation for the inverse discrete Fourier transform be as given in (4.7). Then the ***magnitude of the nth frequency component***, ***A_n***, is given by

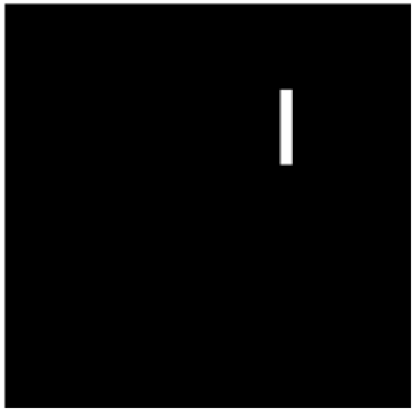
  $$\boldsymbol{A}_n = \sqrt{\boldsymbol{a}_n^2 + \boldsymbol{b}_n^2}\ , \quad 0 \le n \le N-1$$

- The ***phase of the nth frequency component***, $\emptyset_n$, is given by $\emptyset_n = -tan^{-1}\left({}^{\boldsymbol{b}_n}/_{\boldsymbol{a}_n}\right), 0 \le n \le N-1$

- The ***magnitude/phase form of the inverse DFT*** is given by $\boldsymbol{f}_k = \sum_{n=0}^{N-1} \boldsymbol{A}_n \cos(2\pi nk + \emptyset_n)$

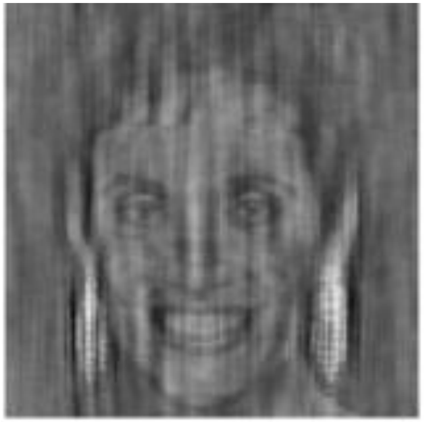
Phase Information

- Phase information in images – important!

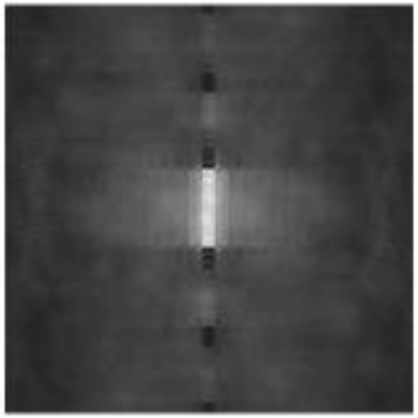


(A)

(B)



(A’s spectrum
B’s phase angle)



(B’s spectrum
A’s phase angle)

Phase Information

- Phase information in audio
  - Audio is a wave that continuously comes into your ears, so actually we don’t detect the phase difference.
  - However, if two waves of the same frequency, one is phase shifted, come to you at the same time, then you will hear the destructive interference.

Fast Fourier Transform(FFT)

- The usefulness of the discrete Fourier transform was extended greatly when a fast version was invented by Cooley and Tukey in 1965. This implementation, called the ***fast Fourier transform*** **(*FFT*)**, reduces the computational complexity from $O(N^2)$ to $O(N \log_2(N))$. N is the number of samples.
- The FFT is efficient because redundant or unnecessary computations are eliminated. For example, there's no need to perform a multiplication with a term that contains sin(0) or cos(0).

FFT

- The FFT algorithm has to operate on blocks of samples where the number of samples is a power of 2.
- The size of the FFT window is significant here because adjusting its size is a tradeoff between frequency and time resolution. You have seen that for an FFT window of size *N*, *N*/2 frequency components are produced. Thus, **the larger the FFT size, the greater the frequency resolution**. However, **the larger the FFT size**, **the smaller the time resolution**. Why?

FFT

- What would happen if the window size doesn’t fit an integer-multiple of the signal’s period?
- For example, Assume that the FFT is operating on 1024 samples of a 440 Hz wave sampled at 8000 samples per second. Then the window contains (1024/8000)*440=56.32 cycles => the end of the window would break the wave in the middle of a cycle.
- Due to this phenomenon(called ***spectral leakage***), the FFT may assume the original signal looks like Fig.4.21.

  The signal becomes discontinuous.

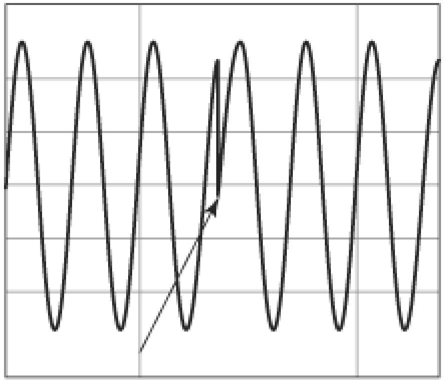
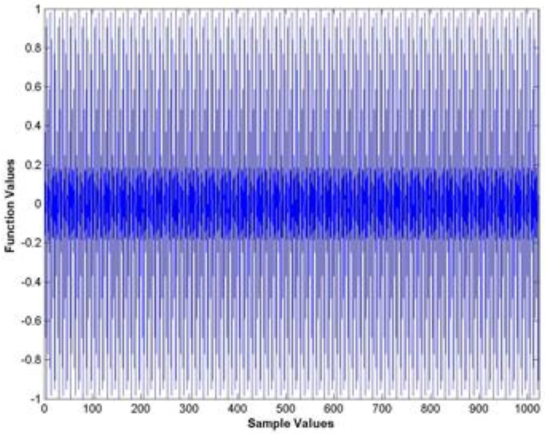
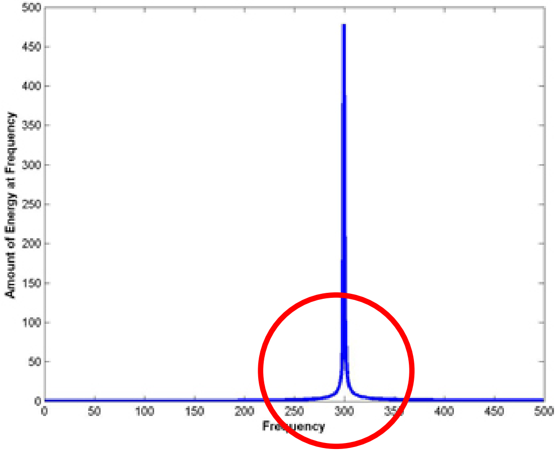


Fig.4.21

Spectral Leakage

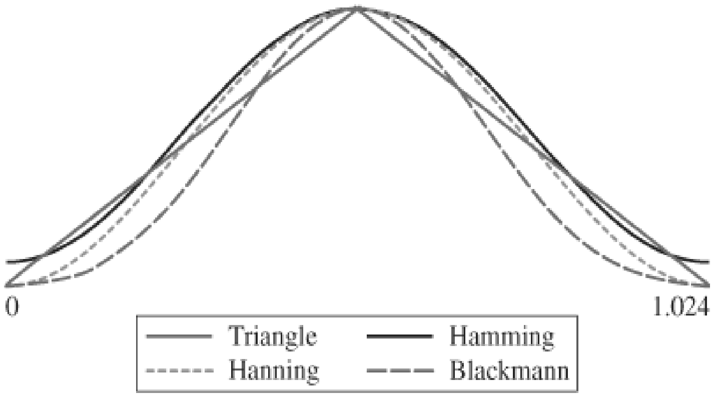
- A simple sinusoidal wave of 300Hz
- After FFT, some frequencies other than 300Hz appear due to spectral leakage





Windowing Function

- **Window function** - to reduce the amplitude of the sound wave at the beginning and end of the FFT window. If the amplitude of the wave is smaller at the beginning and end of the window, then the spurious frequencies will be smaller in magnitude as well.



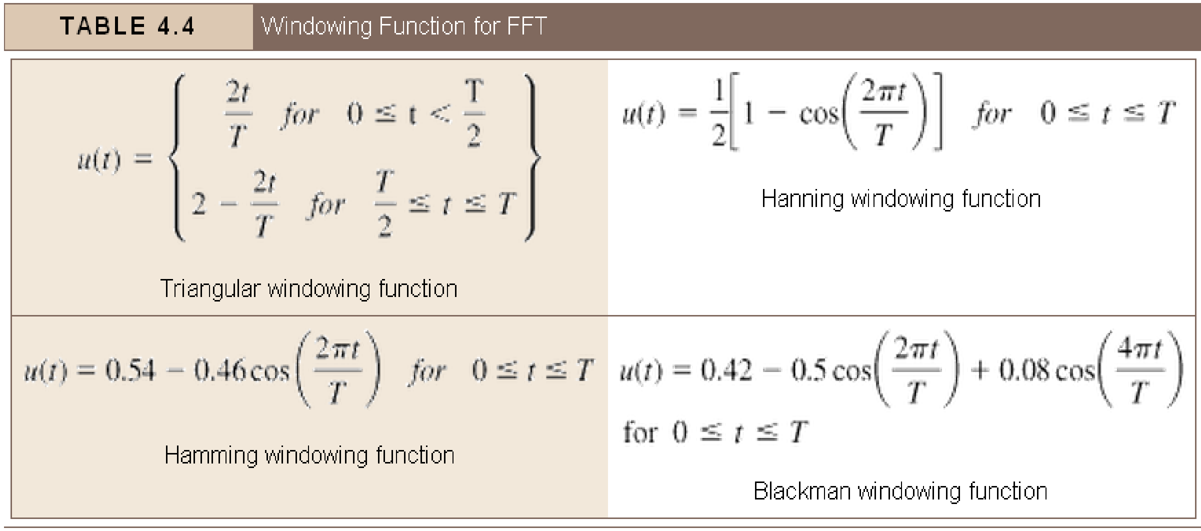
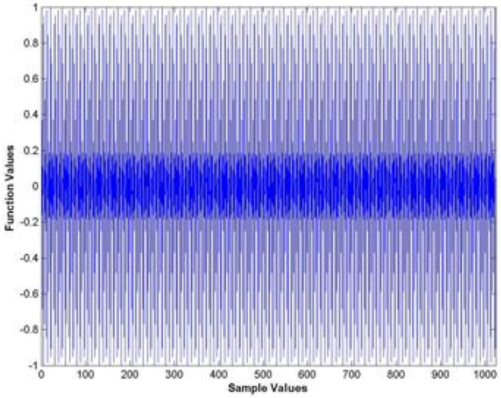


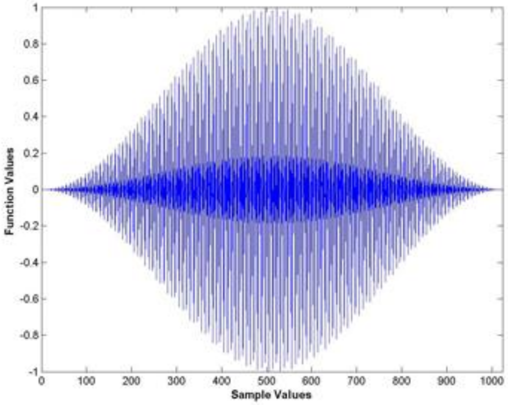
TABLE 4.4	Windowing Function for FFT
$u(t) = \begin{cases} \frac{2t}{T} & \text{for} \quad 0 \leq t < \frac{T}{2} \\ 2 - \frac{2t}{T} & \text{for} \quad \frac{T}{2} \leq t \leq T \end{cases}$ Triangular windowing function	$u(t) = \frac{1}{2}\left[1 - \cos\left(\frac{2\pi t}{T}\right)\right] \quad \text{for} \quad 0 \leq t \leq T$ Hanning windowing function
$u(t) = 0.54 - 0.46\cos\left(\frac{2\pi t}{T}\right) \quad \text{for} \quad 0 \leq t \leq T$ Hamming windowing function	$u(t) = 0.42 - 0.5\cos\left(\frac{2\pi t}{T}\right) + 0.08\cos\left(\frac{4\pi t}{T}\right)$ for $0 \leq t \leq T$ Blackman windowing function

Window Function

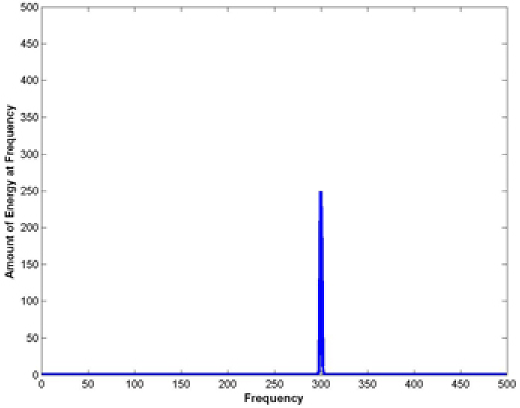
- The frequency components become more accurate, but the magnitudes also decrease. To counteract this, some other algorithms would be used.



Original wave



Applying window function



FFT result

MIDI

- What do you know about MIDI?
  - A kind of music file format like *.wav, *.mp3, *.midi?
  - A kind of music without human voices?
- MIDI is short for Musical Instrument Digital Interface
- Actually, MIDI is far from you can imagine. MIDI is a standard protocol defining how MIDI messages are constructed, transmitted, and stored. These messages communicates between musical instruments and computer softwares.

MIDI vs. Sampled Digital Audio

- A sampled digital audio file contains a vector of samples. These samples are reconstructed into an analog waveform when the audio is played.
- MIDI stores "sound events" or "human performances of sound" rather than sound itself.
- A MIDI file contains messages that indicate the notes, instruments, and duration of notes to be played. In MIDI terminology, each message describes an **event**(the change of note, key, tempo, etc.)

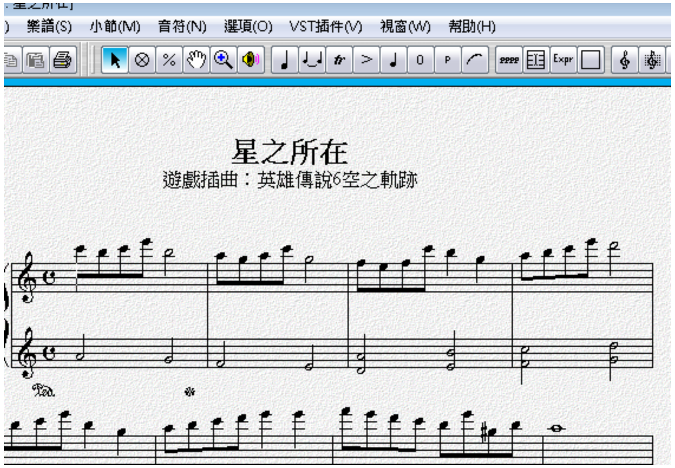
Advantages and Disadvantages

- Advantages
  - Requiring relatively few bytes to store a file compared with sampled audio file, why?
  - Easy to create and edit music
- Disadvantages
  - More artificial and mechanical(sampled audio can capture all the characteristics of the music)

How MIDI Files Are Created, Edited, and Played?

- From hardware
  - When you play the electronic piano with it connected to a PC, midi messages are being created.

- From software
  - Overture



星之所在

遊戲插曲：英雄傳說6空之軌跡

Musical Acoustics and Notation

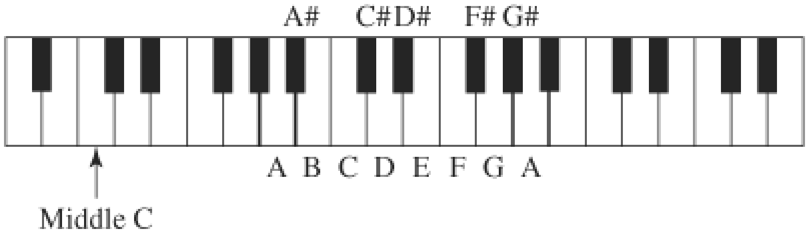
- The range of human hearing is from about 20 Hz to about 20,000 Hz. As you get older, you lose your ability to hear high frequency sounds.
  - Test if you can hear the frequency you should be able to hear at your age
  - http://www.ultrasonic-ringtones.com/
- If the frequency of one note is 2^n times of the frequency of another, where *n* is an integer, the two notes sound "the same" to the human ear, except that the first is higher-pitched than the second.

  (n=1 => 高八度)

Musical Acoustics and Notation

- Let *g* be the frequency of a musical note. Let *h* be the frequency of a musical tone *n* ***octaves*** higher than *g*.
  $\Rightarrow h = 2^n g$

- The word octave comes from the fact that there are eight whole notes(全音)between two notes that sound the same.
- There are 12 notes in an octave:



Musical Acoustics and Notation

- If we know the frequency of a certain note, how to compute the others?

  $2f = f \cdot x^{12} \Rightarrow x \cong 1.059463$

- That is, if A has frequency 440Hz, then A# has frequency 440*1.059463=466.16Hz

Summary

- Audio signal sampling, digitization
- Audio dithering
- Noise shaping
- Non-linear quantization
- Frequency analysis – FFT
- MIDI