



Welcome Back to Fundamentals of Multimedia (MR412)

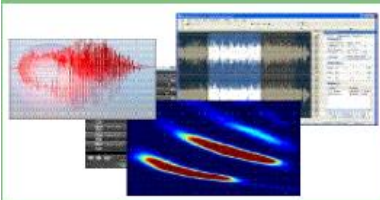
Fall, 2012

Chapter 6

ZHU Yongxin, Winson

zhuyongxin@sjtu.edu.cn





[Home](#)

[Research](#)

[Publications](#)

[Teaching](#)

[Service](#)

Sound and Music Computing Lab

SMC@NUS

- [Main Page](#)
- [Projects](#)
- [Publications](#)
- [People](#)
- [Resource](#)

Image:M3rshow.jpg

[Image](#) [File history](#) [Links](#)



Personalized Multimodal Music Search

Find Your Favourite Music with Less Effort

Powered by



yanini

No file chosen

Personalization

Last Updated on 8th June - 2009



Personalized Multimodal Music Search

Find Your Favourite Music with Less Effort

Powered by



No file chosen

Personalization

- Text
 Genre
 Mood
 Instrument
 Vocal
 Tempo

Yanni - The Storm



Rate: ★★★★★

Views: 2747527

Text Comments

skaterking63 (2009-06-01T02:22:56.000Z)

never think negative think positive im seriopus its working for me

SatoxD

Info

Violin: Samvel Yervinyan [Armenia]; Violin: Sayaka Katsuki [Japan]; Harp: Victor Espinola [Paraguay]; Trumpet: Ramon Flores [Mexico] Yanni Live! The Concert Event.

URL

http://www.youtube.com/v/SSrH_oqnrI0&f=videos&

Related Videos



Yanni - Best Violin Player EVER(best violin solo)
 1116774 views
 Fight3rFX



Yanni - Rainmaker
 1781078 views
 SatoxD



Yanni - For All Seasons
 838726 views
 SatoxD



Personalized Multimodal Music Search

Find Your Favourite Music with Less Effort

Powered by
YouTube

beethoven

Personalization ▾

[Moonlight Sonata](#)

dedicated to Ludwig van Beethoven

★★★★★ 23243578 views [beethovenslady](#)

[Beethoven Symphony No.9](#)

Molto vivace. Leipzig Gewandhaus Orchestra!!

★★★★★ 7444479 views [Mitsumi122](#)

[Beethoven 5th Symphony \(No. 5, graphical score animation, allegro\)](#)

Download?" and other FAQ for Ludwig van Beethoven's 5th Symphony 5, First mvt.,accompanied by a scr...

★★★★★ 1316245 views [smalin](#)

[Beethoven - Fur elise](#)

Ivo Pogorelich plays Beethoven's "Fur elise". Video messes up in some spots, but it's nothing too b...

★★★★★ 11701205 views [mattzart](#)

[Beethoven's 5th Symphony](#)

This is the first movement of Beethoven's 5th symphony. Composed between 1804 and 1808.

★★★★★ 1611696 views [castout888](#)

[Wilhelm Kempff plays Beethoven's Moonlight Sonata mvt. 1](#)

Wilhelm Kempff plays Beethoven's Moonlight Sonata mvt. 1

★★★★★ 3537039 views [popololopolo](#)

[Wilhelm Kempff plays Beethoven's Moonlight Sonata mvt. 3](#)

Wilhelm Kempff plays Beethoven's Moonlight Sonata mvt. 3

★★★★★

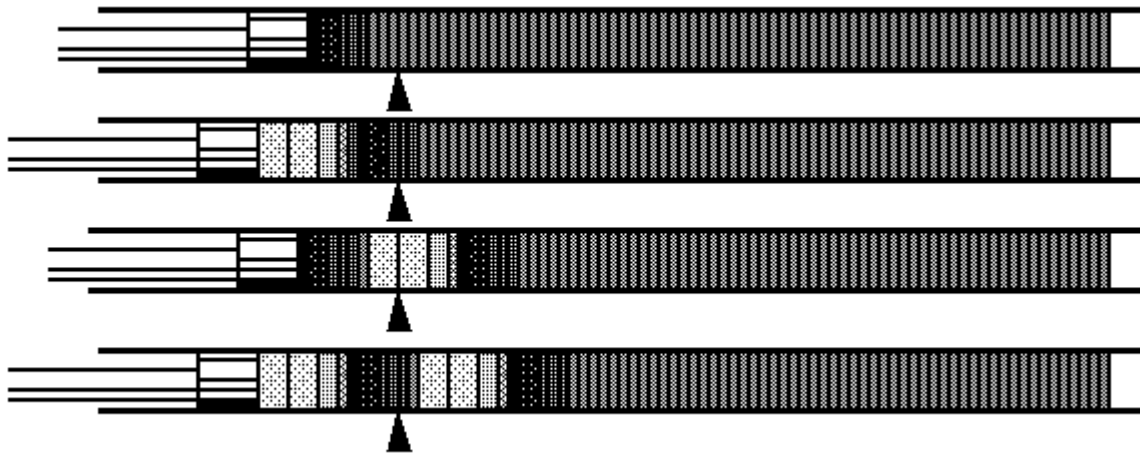


- [6.1 Audio Signals](#)
- [6.2 Quantization of Audio](#)
- [6.3 The MIDI Concepts](#)
- [6.4 Transformation of Audio for Transmission](#)



Facts about Sound

Sound is detected by measuring the pressure level at a point:

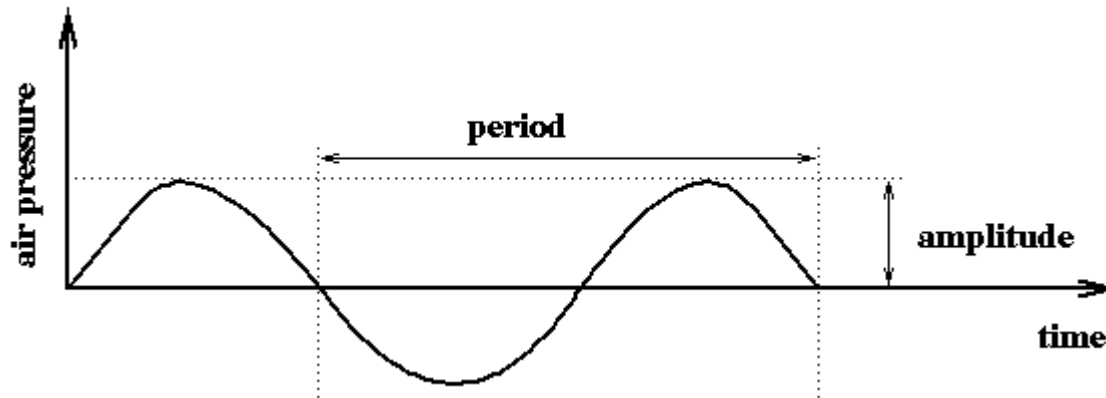


Source: -- Brian Smith notes, 1996.



Facts about Sound

- The amplitude of a sound is the measure of displacement of the air pressure wave from its mean:



Source: -- Brian Smith notes, 1996.



6.1 Audio Signals

What is Sound?

- Sound is a wave phenomenon like light, but is macroscopic and involves molecules of air being compressed and expanded under the action of some physical device.
- (a) For example, a speaker in an audio system vibrates back and forth and produce *longitudinal* pressure wave that we perceive as sound.
- (b) Since sound is a pressure wave, it takes on continuous values, as opposed to digitized ones.



- (c) Even though such pressure waves are longitudinal, they still have ordinary wave properties and behaviors, such as reflection (bouncing), refraction (change of angle when entering a medium with a different density) and direction (bending around an obstacle).
- (d) If we wish to use a digital version of sound waves we must form digitized representations of audio information.



Digitization



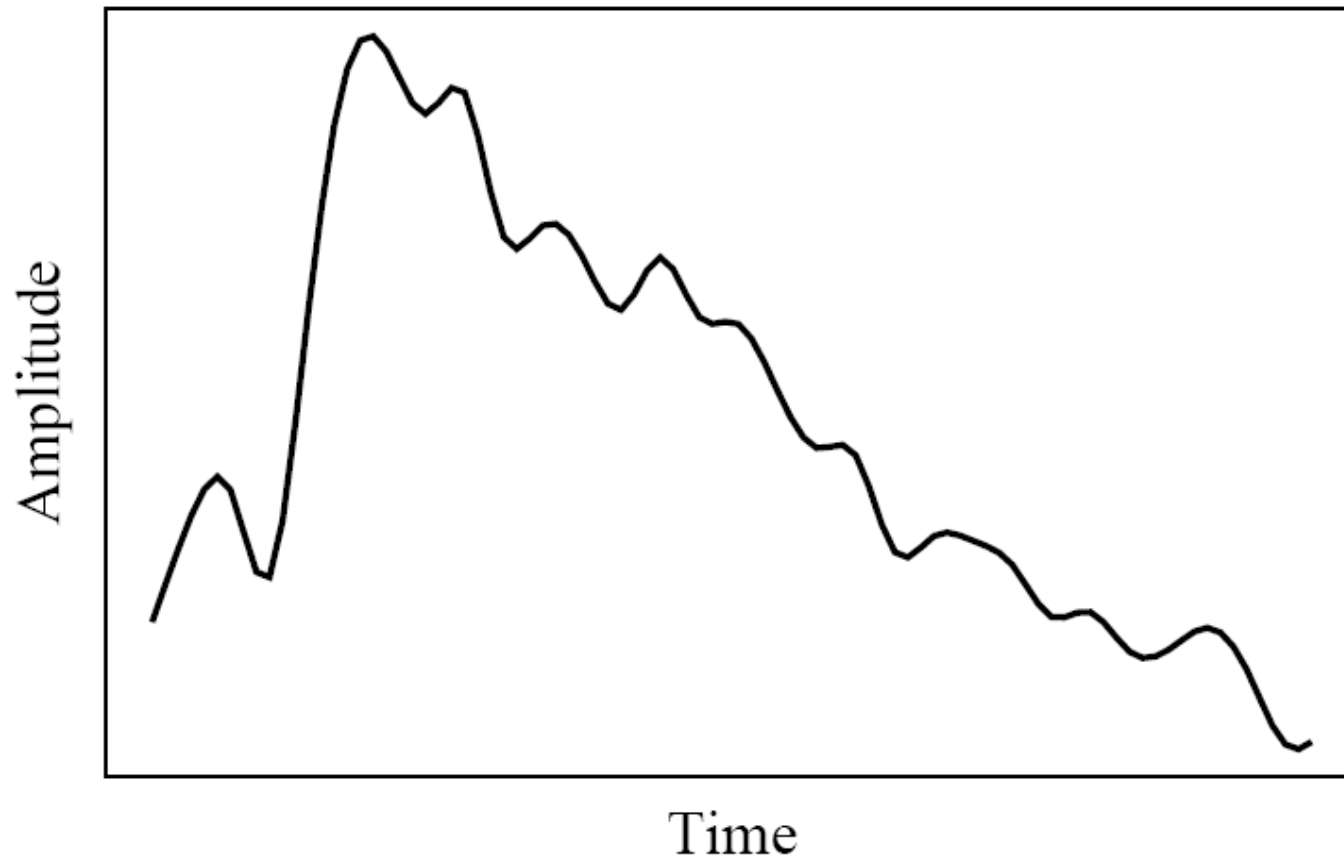

-  **Digitization** means conversion to a stream of numbers, and preferably these numbers should be integers for efficiency.
-  Fig. 6.1 shows the 1-dimensional nature of sound: **amplitude** values depend on a 1D variable, time. (And note that images depend instead on a 2D set of variables, x and y).



Fig. 6.1: An analog signal: continuous measurement of pressure wave.





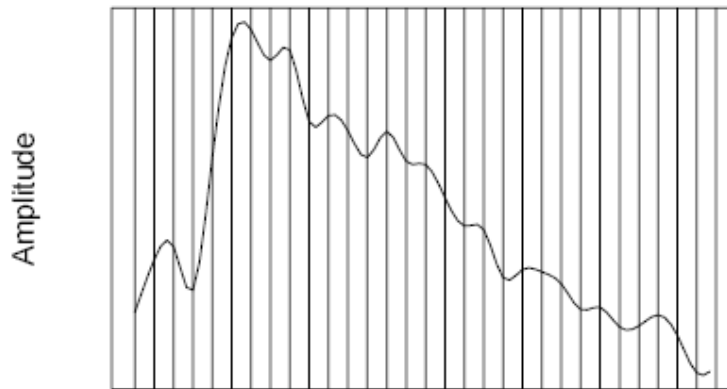
 The graph in Fig. 6.1 has to be made digital in both time and amplitude. To digitize, the signal must be **sampled** in each dimension: in time, and in amplitude.

- (a) Sampling means measuring the quantity we are interested in, usually at evenly-spaced intervals.
- (b) The 1st kind of sampling, using measurements only at evenly spaced time intervals, is simply called, *sampling*. The rate at which it is performed is called the *sampling frequency*.
- (c) For audio, typical sampling rates are from 8 kHz (8,000 samples per second) to 48 kHz. This range is determined by Nyquist theorem discussed later.
- (d) Sampling in the amplitude or voltage dimension is called **quantization**.

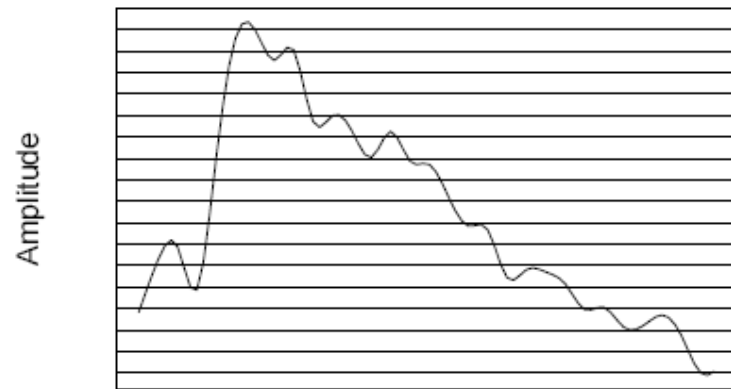


Sampling and Quantization

- (a): Sampling the analog signal in the time dimension.
- (b): Quantization is sampling the analog signal in the amplitude dimension.



Time
(a)



Time
(b)



Sampling decisions



Thus to decide how to digitize audio data we need to answer the following questions:

- 1. What is the sampling rate?
- 2. How finely is the data to be quantized, and is quantization uniform?
- 3. How is audio data formatted? (file format)



Nyquist Theorem

- Signals can be decomposed into a sum of sinusoids. Fig. 6.3 shows how weighted sinusoids can build up quite a complex signal.



Building up a complex signal by superposing sinusoids

Fundamental
frequency



+ 0.5 ×
2 × fundamental



=



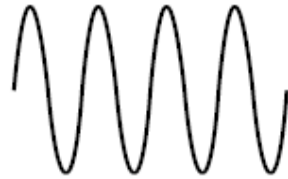
+ 0.33 ×
3 × fundamental



=



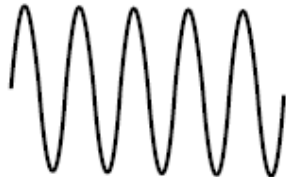
+ 0.25 ×
4 × fundamental



=



+ 0.5 ×
5 × fundamental



=





Nyquist Theorem

- Whereas **frequency** is an absolute measure, **pitch** is generally relative -- a perceptual subjective quality of sound.
- (a) Pitch and frequency are linked by setting the note A above middle C to exactly 440 Hz.
- (b) An **octave** above that note takes us to another A note. An octave corresponds to *doubling the frequency*. Thus with the middle “A” on a piano (“A4” or “A440”) set to 440 Hz, the next “A” up is at 880 Hz, or one octave above.
- (c) **Harmonics**: any series of musical tones whose frequencies are integral multiples of the frequency of a fundamental tone: Fig. 6.
- (d) If we allow non-integer multiples of the base frequency, we allow non-“A” notes and have a more complex resulting sound.

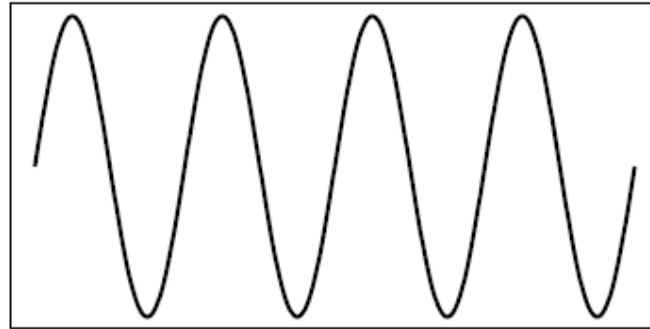


Nyquist Theorem

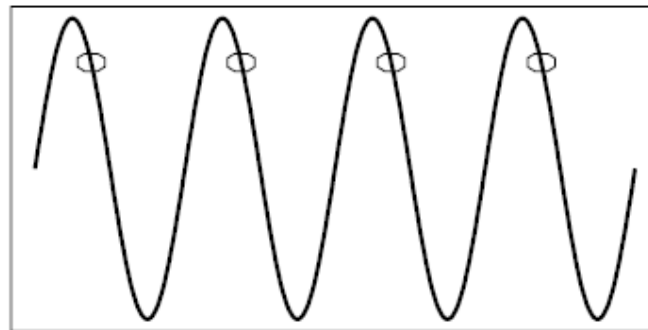
- ④ The Nyquist theorem states how frequently we must sample in time to be able to recover the original sound.
- ④ (a) Fig. 6.4(a) shows a single sinusoid: it is a single, pure, frequency (only electronic instruments can create such sounds).
- ④ (b) If sampling rate just equals the actual frequency, Fig. 6.4(b) shows that a false signal is detected: it is simply a constant, with zero frequency.
- ④ (c) Now if sample at 1.5 times the actual frequency, Fig. 6.4(c) shows that we obtain an incorrect (**alias**) frequency that is lower than the correct one | it is half the correct one (the wavelength, from peak to peak, is double that of the actual signal).
- ④ (d) Thus for correct sampling we must use a sampling rate equal to at least *twice the maximum frequency* content in the signal. This rate is called the **Nyquist rate**.



Nyquist Theorem



(a)



(b)

Signal to Quantization Noise Ratio (SQNR)

- ④ Aside from any noise that may have been present in the original analog signal, there is also an additional error that results from quantization.
 - (a) If voltages are actually in 0 to 1 but we have only 8 bits in which to store values, then effectively we force all continuous values of voltage into only 256 different values.
 - (b) This introduces a round-off error. It is not really “noise”. Nevertheless it is called **quantization noise** (or quantization error).

Signal to Quantization Noise Ratio (SQNR)

- ④ The quality of the quantization is characterized by the Signal to Quantization Noise Ratio (**SQNR**).
- (a) **Quantization noise**: the difference between the actual value of the analog signal, for the particular sampling time, and the nearest quantization interval value.
- (b) At most, this error can be as much as half of the interval.

Signal to Quantization Noise Ratio (SQNR)

- ⊗ (c) For a quantization accuracy of N bits per sample, the SQNR can be simply expressed:

$$\begin{aligned} SQNR &= 20 \log_{10} \frac{V_{signal}}{V_{quan_noise}} = 20 \log_{10} \frac{2^{N-1}}{\frac{1}{2}} \\ &= 20 \times N \times \log 2 = 6.02 N(\text{dB}) \end{aligned} \quad (6.3)$$

⊗ Notes:

- (a) We map the maximum signal to $2^{N-1} - 1 \sim (2^{N-1})$ and the most negative signal to -2^{N-1} .
- (b) Eq. (6.3) is the *Peak* signal-to-noise ratio, PSQNR: peak signal and peak noise.



Linear and Non-linear Quantization

- ⊗ **Linear format:** samples are typically stored as uniformly quantized values.
- ⊗ **Non-uniform quantization:** set up more nearly-spaced levels where humans hear with the most acuity.
 - Weber's Law stated formally says that equally perceived differences have values proportional to absolute levels:
$$\Delta \text{Response} \propto \Delta \text{Stimulus} / \text{Stimulus} \quad (6:5)$$
 - Inserting a constant of proportionality k , we have a differential equation that states:
$$dr = k (1/s) ds \quad (6:6)$$
with response r and stimulus s .



Integrating, we arrive at a solution

$$r = k \ln s + C \quad (6:7) \text{ with constant of integration } C.$$

Stated differently, the solution is

$$r = k \ln(s/s_0) \quad (6:8)$$

s_0 = the lowest level of stimulus that causes a response
($r = 0$ when $s = s_0$).



Nonlinear quantization works by first transforming an analog signal from the raw s space into the theoretical r space, and then uniformly quantizing the resulting values.



Such a law for audio is called **μ -law** encoding, (or **u-law**). A very similar rule, called **A-law**, is used in telephony in Europe.



The equations for these very similar encodings are as follows:



μ -law and A-law

μ -law:

$$r = \frac{\text{sgn}(s)}{\ln(1 + \mu)} \ln \left\{ 1 + \mu \left| \frac{s}{s_p} \right| \right\}, \quad \left| \frac{s}{s_p} \right| \leq 1 \quad (6.9)$$

A-law:

$$r = \begin{cases} \frac{A}{1 + \ln A} \left(\frac{s}{s_p} \right), & \left| \frac{s}{s_p} \right| \leq \frac{1}{A} \\ \frac{\text{sgn}(s)}{1 + \ln A} \left[1 + \ln A \left| \frac{s}{s_p} \right| \right], & \frac{1}{A} \leq \left| \frac{s}{s_p} \right| \leq 1 \end{cases} \quad (6.10)$$

$$\text{where } \text{sgn}(s) = \begin{cases} 1 & \text{if } s > 0, \\ -1 & \text{otherwise} \end{cases}$$

- Fig. 6.6 shows these curves. The parameter μ is set to $\mu = 100$ or $\mu = 255$; the parameter A for the A-law encoder is usually set to $A = 87.6$.



μ -law and A-law

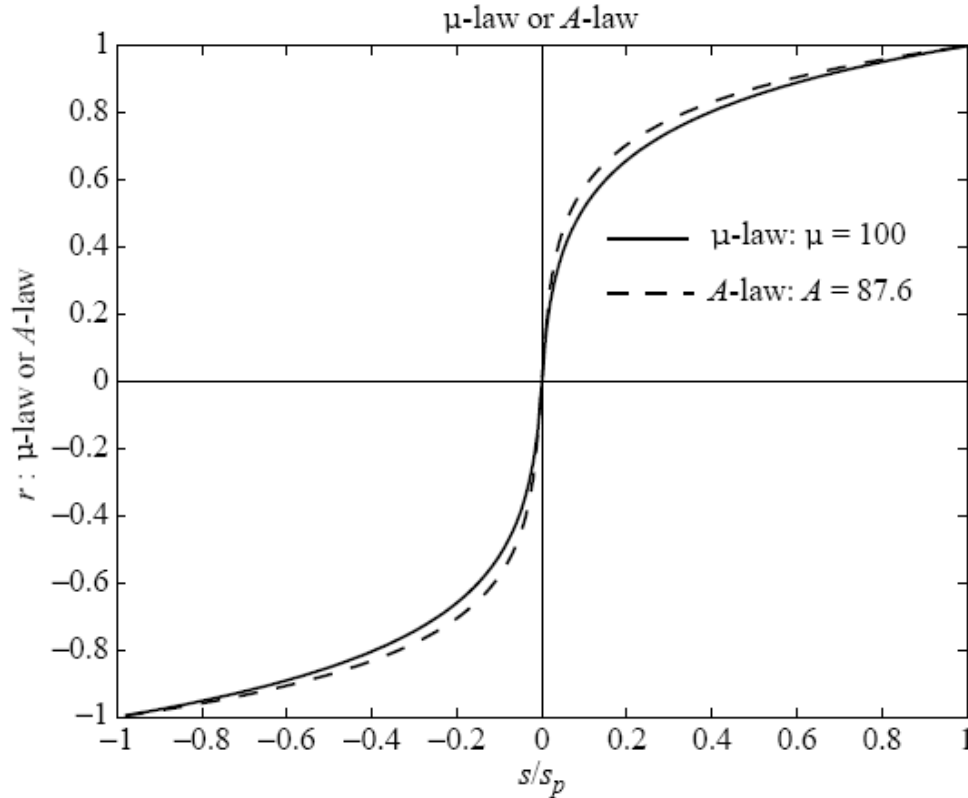


Fig. 6.6: Nonlinear transform for audio signals

- The μ -law in audio is used to develop a nonuniform quantization rule for sound: uniform quantization of r gives finer resolution in s at the quiet end.



6.3 MIDI: Musical Instrument Digital Interface

- Use the sound card's defaults for sounds -> use a simple scripting language and hardware setup called **MIDI**.

- **MIDI Overview**

(a) MIDI is a scripting language | it codes “events” that stand for the production of sounds. E.g., a MIDI event might include values for the pitch of a single note, its duration, and its volume.

(b) MIDI is a standard adopted by the electronic music industry for controlling devices, such as synthesizers and sound cards, that produce music.



- (c) The MIDI standard is supported by most synthesizers, so sounds created on one synthesizer can be played and manipulated on another synthesizer and sound reasonably close.
- (d) Computers must have a special MIDI interface, but this is incorporated into most sound cards. The sound card must also have both D/A and A/D converters.



MIDI channels are used to separate messages.

- (a) There are 16 channels numbered from 0 to 15. The channel forms the last 4 bits (the least significant bits) of the message.
- (b) Usually a channel is associated with a particular instrument: e.g., channel 1 is the piano, channel 10 is the drums, etc.
- (c) Nevertheless, one can switch instruments midstream, if desired, and associate another instrument with any channel.



- **System messages**
 - (a) Several other types of messages, e.g. a general message for all instruments indicating a change in tuning or timing.
 - (b) If the first 4 bits are all 1s, then the message is interpreted as a **system common message**.
- The way a synthetic musical instrument responds to a MIDI message is usually by simply ignoring any play sound message that is not for its channel.
 - If several messages are for its channel, then the instrument responds, provided it is multi-voice, i.e., can play more than a single note at once.



- It is easy to confuse the term **voice with the term timbre**, the latter is MIDI terminology for just what instrument that is trying to be emulated, e.g. a piano as opposed to a violin it is the quality of the sound.
 - (a) An instrument (or sound card) that is **multi-timbral is one that is** capable of playing many different sounds at the same time, e.g., piano, brass, drums, etc.
 - (b) On the other hand, the term **voice, while sometimes used by musicians** to mean the same thing as timbre, is used in MIDI to mean every different timbre and pitch that the tone module can produce at the same time.
- Different timbres are produced digitally by using a **patch** - the set of control settings that define a particular timbre. Patches are often organized into databases, called **banks**.



General MIDI: A standard mapping specifying what instruments (what patches) will be associated with what channels.

- (a) In General MIDI, channel 10 is reserved for percussion instruments, and there are 128 patches associated with standard instruments.
- (b) For most instruments, a typical message might be a Note On message (meaning, e.g., a keypress and release), consisting of what channel, what pitch, and what “velocity” (i.e., volume).
- (c) For percussion instruments, however, the pitch data means which kind of drum.
- (d) A Note On message consists of “status” byte - which channel, what pitch - followed by two data bytes. It is followed by a Note Off message, which also has a pitch (which note to turn off) and a velocity (often set to zero).

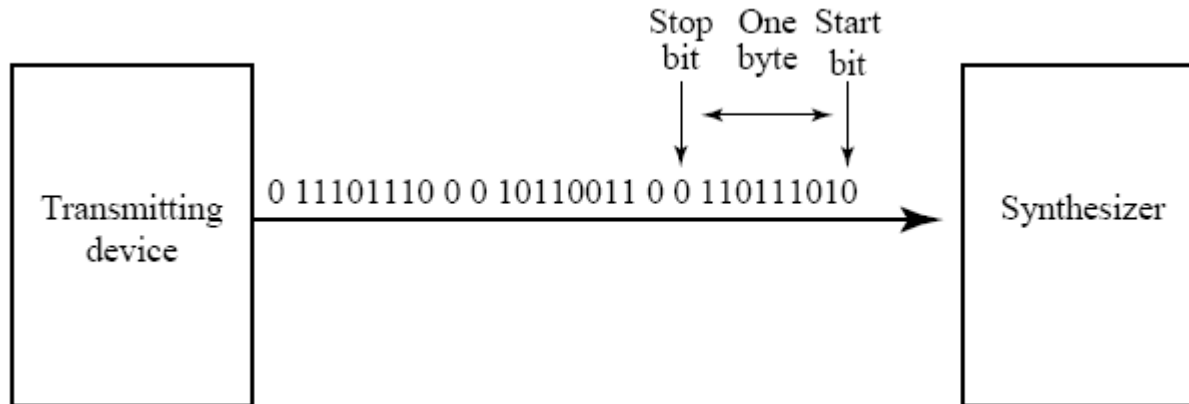


上海交通大学

Shanghai Jiao Tong University



- The data in a MIDI status byte is between 128 and 255; each of the data bytes is between 0 and 127. Actual MIDI bytes are 10-bit, including a 0 start and 0 stop bit.



- Fig. 6.8: Stream of 10-bit bytes; for typical MIDI messages, these consist of $\{Status\ byte, Data\ Byte, Data\ Byte\} = \{Note\ On, Note\ Number, Note\ Velocity\}$



- A MIDI device often is capable of **programmability**, and also can change the **envelope** describing how the amplitude of a sound changes over time.
- Fig. 6.9 shows a model of the response of a digital instrument to a Note On message:

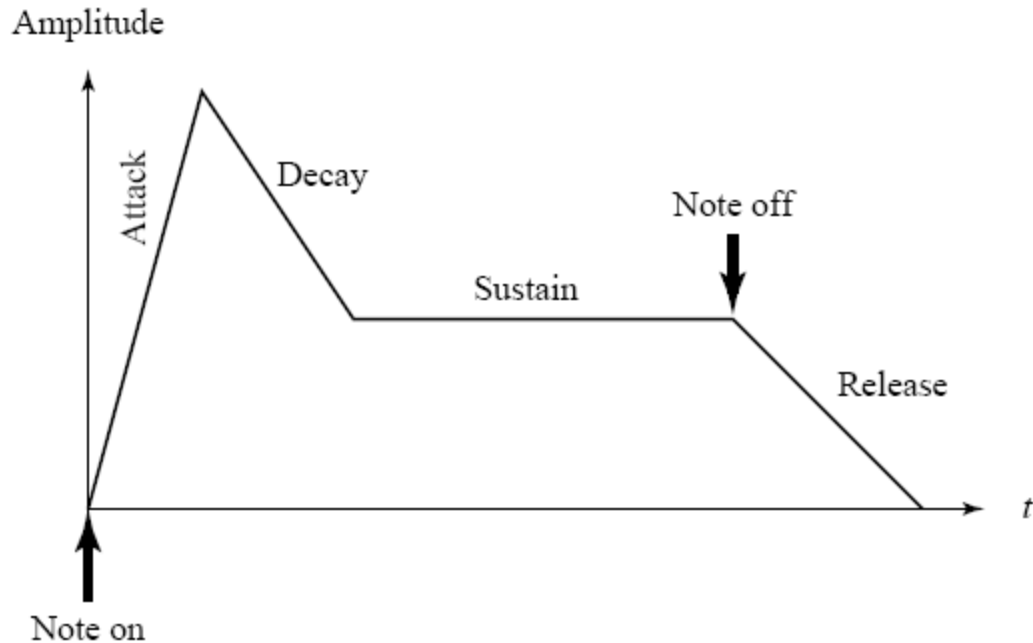


Fig. 6.9: Stages of amplitude versus time for a music note



Hardware Aspects of MIDI

- The MIDI hardware setup consists of a 31.25 kbps serial connection. Usually, MIDI-capable units are either Input devices or Output devices, not both.
- A traditional synthesizer is shown in Fig. 6.10:

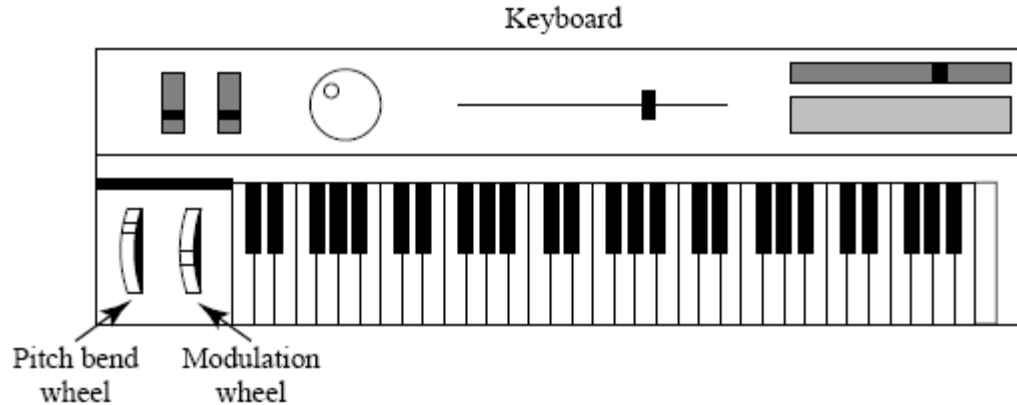


Fig. 6.10: A MIDI synthesizer



The physical MIDI ports consist of 5-pin connectors for IN and OUT, as well as a third connector called THRU.

- (a) MIDI communication is half-duplex.
- (b) MIDI IN is the connector via which the device receives all MIDI data.
- (c) MIDI OUT is the connector through which the device transmits all the MIDI data it generates itself.
- (d) MIDI THRU is the connector by which the device echoes the data it receives from MIDI IN. Note that it is only the MIDI IN data that is echoed by MIDI THRU | all the data generated by the device itself is sent via MIDI OUT.



A typical MIDI sequencer setup is shown in Fig. 6.11:

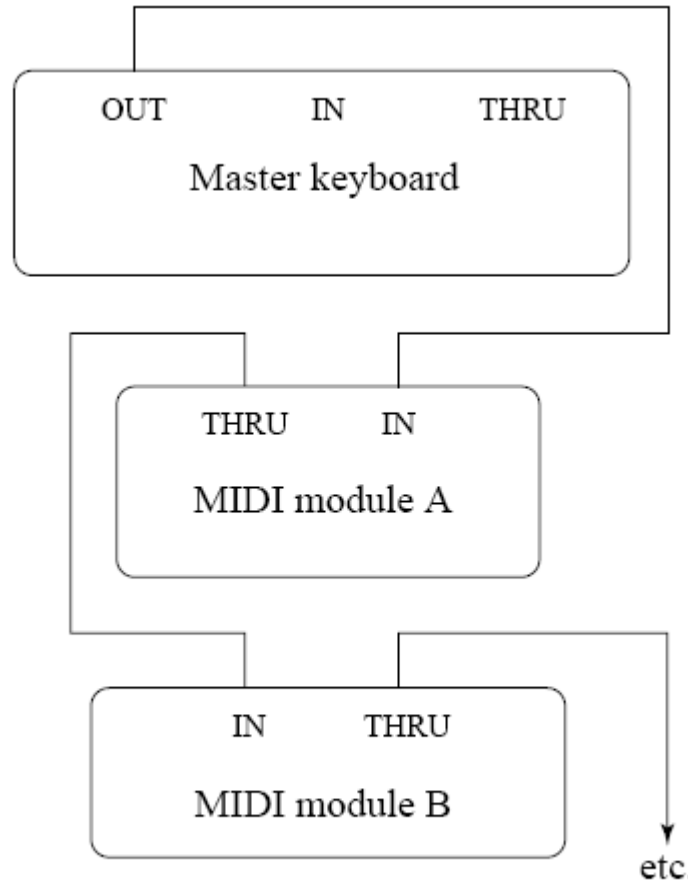


Fig. 6.11: A typical MIDI setup



6.4 Transformation of Audio



Coding of Audio: Quantization and transformation of data are collectively known as **coding** of the data.

- a) For audio, the μ or A-law technique for companding audio signals is usually combined with an algorithm that exploits the temporal redundancy present in audio signals.
- b) Differences in signals between the present and a past time can reduce the size of signal values and also concentrate the histogram of pixel values (differences, now) into a much smaller range.

(companding = compressing and expanding)





Coding of Audio

c) The result of reducing the variance of values is that lossless compression methods produce a bitstream with shorter bit lengths for more likely values (-> expanded discussion in Chap.7).

In general, producing quantized sampled output for audio is called **PCM** (Pulse Code Modulation). The differences version is called **DPCM** (Differential Pulse Code Modulation) and a crude but efficient variant is called **DM (delta modulation)**. The adaptive version is called **ADPCM** (Adaptive Differential Pulse Code Modulation).



Pulse Code Modulation

-  The basic techniques for creating digital signals from analog signals are **sampling** and **quantization**.
-  Quantization consists of selecting breakpoints in magnitude, and then re-mapping any value within an interval to one of the representative output levels. → Repeat of Fig. 6.2:



Pulse Code Modulation

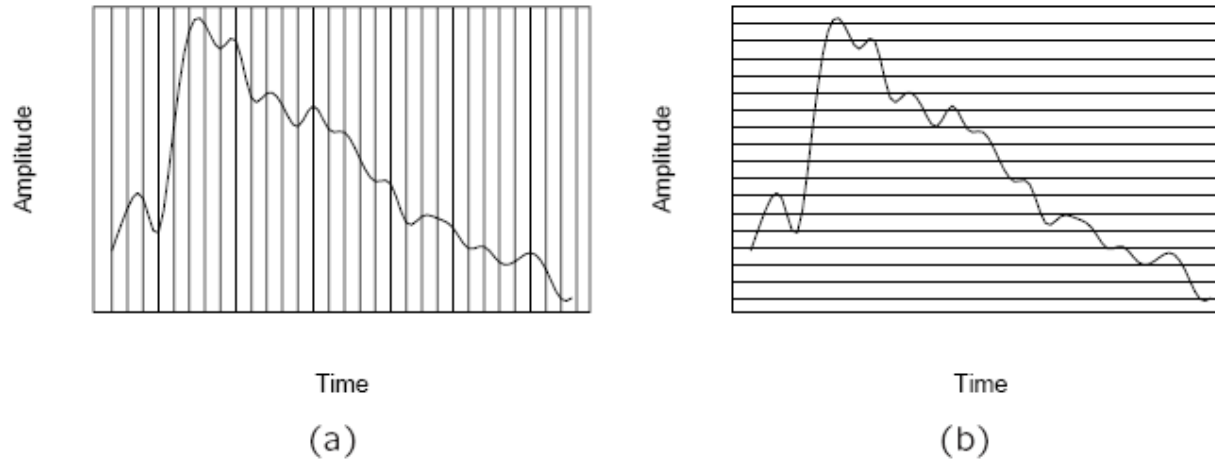






Fig. 6.2: Sampling and Quantization.



Pulse Code Modulation

-  a) The set of interval boundaries are called **decision boundaries**, and the representative values are called **reconstruction levels**.
-  b) The boundaries for quantizer input intervals that will all be mapped into the same output level form a **coder mapping**.
-  c) The representative values that are the output values from a quantizer are a **decoder mapping**.
-  d) Finally, we may wish to **compress** the data, by assigning a bit stream that uses fewer bits for the most prevalent signal values (Chap. 7).



Pulse Code Modulation

- ④ Every compression scheme has three stages:
- ④ A. The input data is **transformed** to a new representation that is easier or more efficient to compress.
- ④ B. We may introduce **loss** of information. *Quantization* is the main lossy step) we use a limited number of reconstruction levels, fewer than in the original signal.
- ④ C. **Coding**. Assign a codeword (thus forming a binary bitstream) to each output level or symbol. This could be a fixed-length code, or a variable length code such as Human coding (Chap. 7).



Pulse Code Modulation



For audio signals, we first consider PCM for digitization. This leads to Lossless Predictive Coding as well as the DPCM scheme; both methods use *differential coding*. As well, we look at the adaptive version, ADPCM, which can provide better compression.



PCM in Speech Compression

- ④ Assuming a bandwidth for speech from about 50 Hz to about 10 kHz, the Nyquist rate would dictate a sampling rate of 20 kHz.
- ④ (a) Using uniform quantization without companding, the minimum sample size we could get away with would likely be about 12 bits. Hence for mono speech transmission the bit-rate would be 240 kbps.
- ④ (b) With companding, we can reduce the sample size down to about 8 bits with the same perceived level of quality, and thus reduce the bit-rate to 160 kbps.
- ④ (c) However, the standard approach to telephony in fact assumes that the highest-frequency audio signal we want to reproduce is only about 4 kHz. Therefore the sampling rate is only 8 kHz, and the companded bit-rate thus reduces this to 64 kbps.



PCM in Speech Compression

- However, there are two small wrinkles we must also address:
- 1. Since only sounds up to 4 kHz are to be considered, all other frequency content must be noise. Therefore, we should remove this high-frequency content from the analog input signal. This is done using a band-limiting filter that blocks out high, as well as very low, frequencies.

Also, once we arrive at a pulse signal, such as that in Fig 6.13(a) below, we must still perform DA conversion and then construct a digital output analog signal. But, effectively, the signal we arrive at is the staircase shown in Fig. 6.13(b).



PCM in Speech Compression

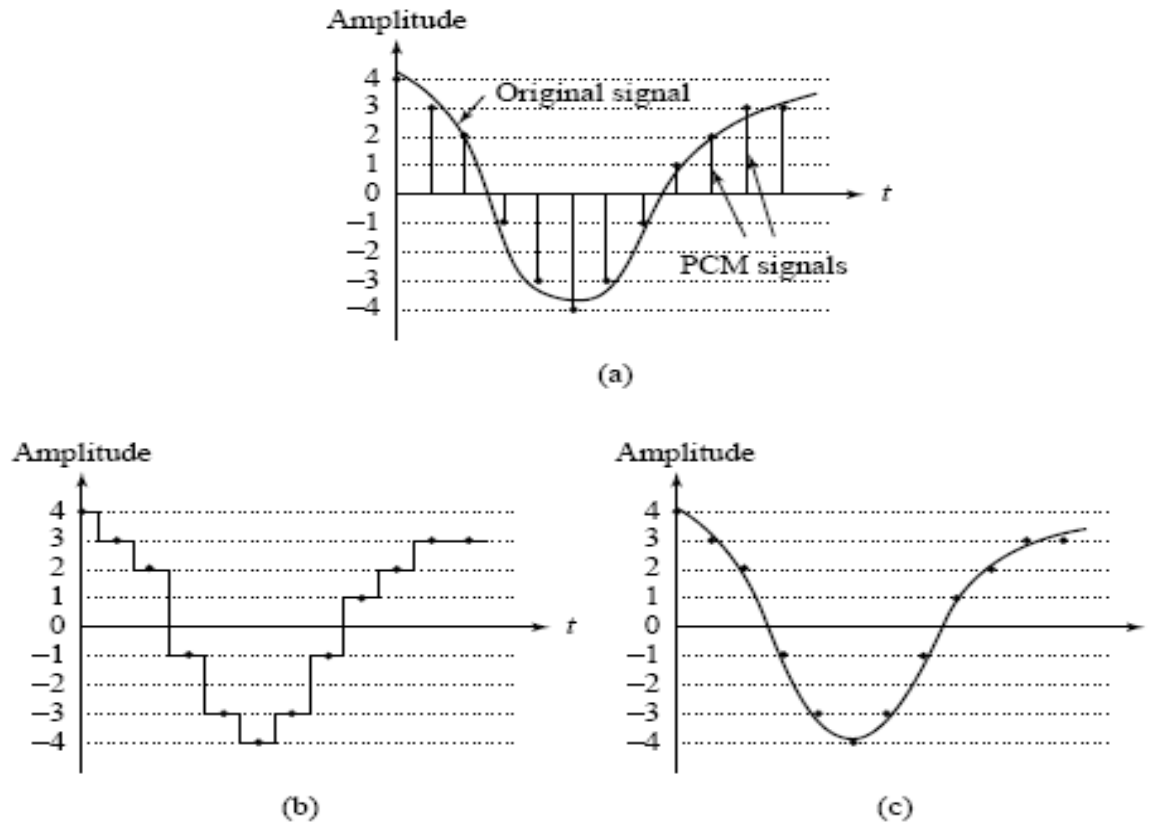


Fig. 6.13: Pulse Code Modulation (PCM). (a) Original analog signal and its corresponding PCM signals. (b) Decoded staircase signal. (c) Reconstructed signal after low-pass filtering.



PCM in Speech Compression

- 2. A discontinuous signal contains not just frequency components due to the original signal, but also a theoretically infinite set of higher-frequency components:
 - (a) This result is from the theory of **Fourier analysis**, in signal processing.
 - (b) These higher frequencies are **extraneous**.
 - (c) Therefore the output of the digital-to-analog converter goes to a **low-pass filter** that allows only frequencies up to the original maximum to be retained.



PCM in Speech Compression

The complete scheme for encoding and decoding telephony signals is shown as a schematic in Fig. 6.14. As a result of the low-pass filtering, the output becomes smoothed and Fig. 6.13(c) above showed this effect.

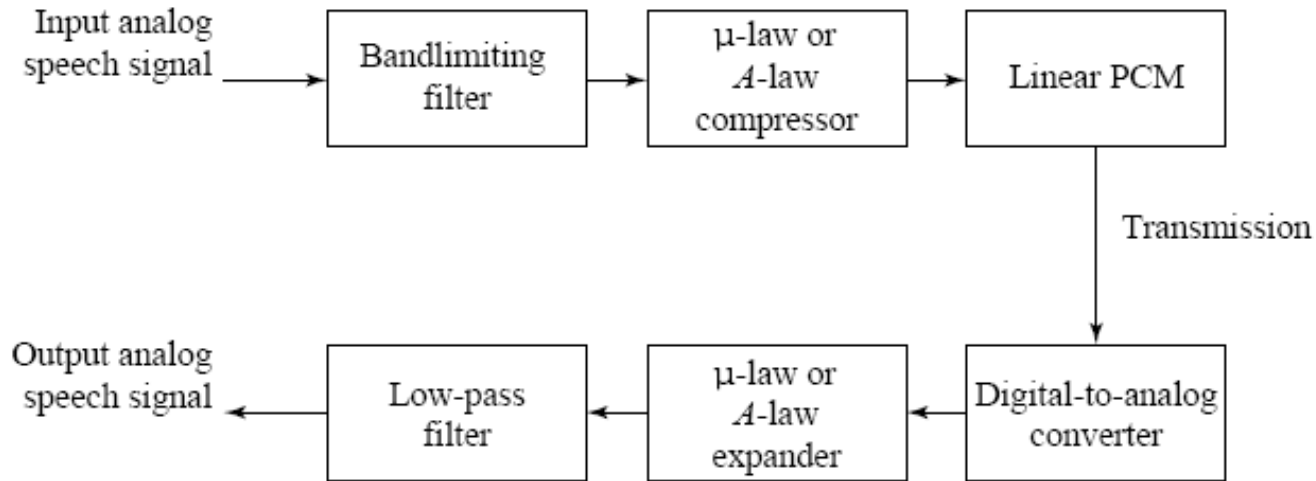



Fig. 6.14: PCM signal encoding and decoding.





Differential Coding of Audio

 Audio is often stored not in simple PCM but instead in a form that exploits differences $|$ which are generally smaller numbers, so offer the possibility of using fewer bits to store.

(a) If a time-dependent signal has some consistency over time ("temporal redundancy"), the difference signal, subtracting the current sample from the previous one, will have a more peaked histogram, with a maximum around zero.



Differential Coding of Audio

-  (b) For example, as an extreme case the histogram for a linear ramp signal that has constant slope is flat, whereas the histogram for the derivative of the signal (i.e., the differences, from sampling point to sampling point) consists of a spike at the slope value.
-  (c) So if we then go on to assign bit-string codewords to differences, we can assign short codes to prevalent values and long codewords to rarely occurring ones.



Lossless Predictive Coding

- ① **Predictive coding:** simply means transmitting differences
 - predict the next sample as being equal to the current sample; send not the sample itself but the difference between previous and next.
- ② (a) Predictive coding consists of finding differences, and transmitting these using a PCM system.
- ③ (b) Note that differences of integers will be integers. Denote the integer input signal as the set of values f_n . Then we **predict** values \hat{f}_n as simply the previous value, and denote the error e_n as the difference between the actual and the predicted signal:

$$\begin{aligned}\hat{f}_n &= f_{n-1} \\ e_n &= f_n - \hat{f}_n\end{aligned}\quad (6.12)$$



- (c) But it is often the case that some **function** of a few of the previous values, f_{n-1} , f_{n-2} , f_{n-3} , etc., provides a better prediction. Typically, a linear **predictor** function is used:

$$\hat{f}_n = \sum_{k=1}^{2 \text{ to } 4} a_{n-k} f_{n-k} \quad (6.13)$$



Lossless Predictive Coding



The idea of forming differences is to make the histogram of sample values more peaked.

- (a) For example, Fig.6.15(a) plots 1 second of sampled speech at 8 kHz, with magnitude resolution of 8 bits per sample.
- (b) A histogram of these values is actually centered around zero, as in Fig. 6.15(b).
- (c) Fig. 6.15(c) shows the histogram for corresponding speech signal **differences**: difference values are much more clustered around zero than are sample values themselves.
- (d) As a result, a method that assigns short codewords to frequently occurring symbols will assign a short code to zero and do rather well: such a coding scheme will much more efficiently code sample differences than samples themselves.



Lossless Predictive Coding

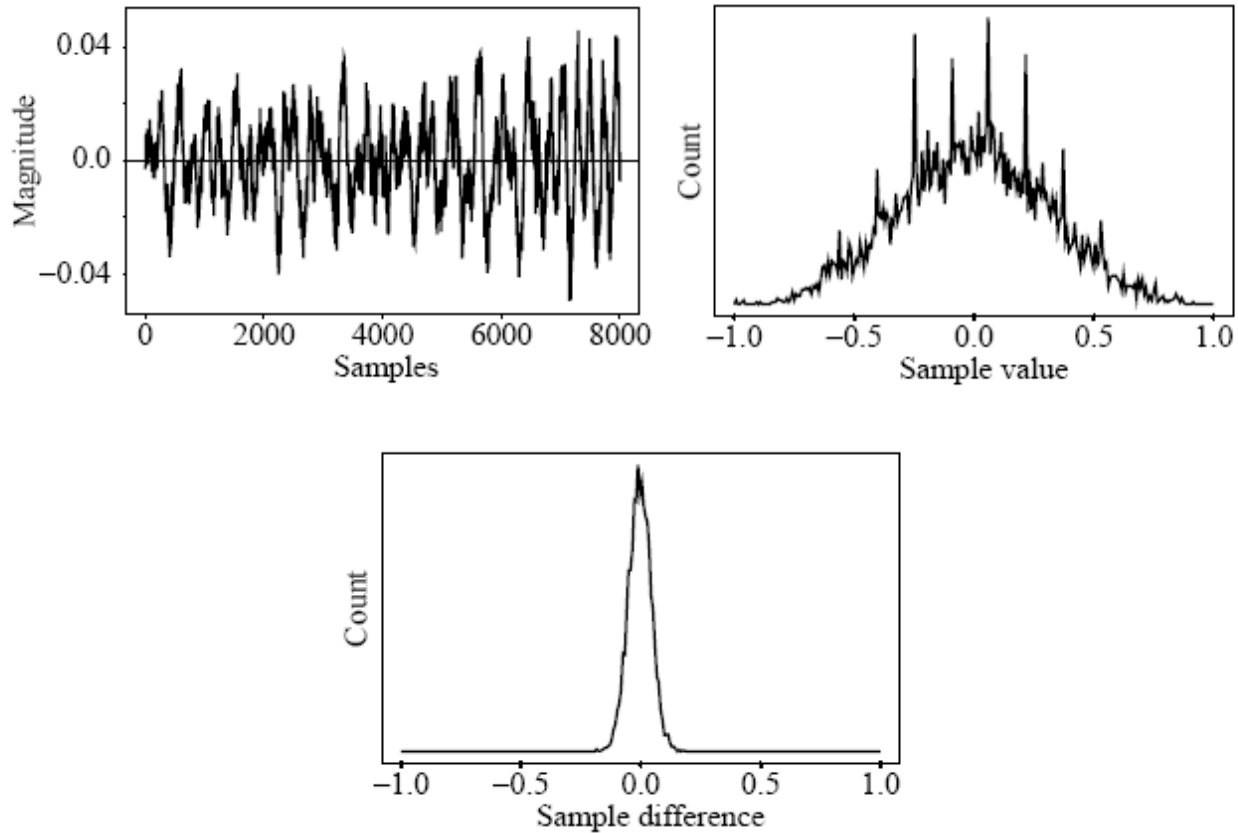


Fig. 6.15: Differencing concentrates the histogram. (a): Digital signal. (b): Histogram of digital speech signal values. (c): Histogram digital speech signal differences.



Lossless Predictive Coding



One problem: suppose our integer sample values are in the range $0..255$. Then differences could be as much as $-255..255$ --- we've increased our **dynamic range** (ratio of maximum to minimum) by a factor of two -> need more bits to transmit some differences.

- (a) A clever solution for this: denote two new codes, denoted SU and SD, standing for Shift-Up and Shift-Down. Some special code values will be reserved for these.
- (b) Then we can use codewords for only a limited set of signal differences, say only the range $-15..16$. Differences which lie in the limited range can be coded as is, but with the extra two values for SU, SD, a value outside the range $-15..16$ can be transmitted as a series of shifts, followed by a value that is indeed inside the range $-15::16$.
- (c) For example, 100 is transmitted as: SU, SU, SU, 4, where (the codes for) SU and for 4 are what are transmitted (or stored).



Lossless Predictive Coding

- Lossless predictive coding -- the decoder produces the same signals as the original. As a simple example, suppose we devise a predictor for \hat{f}_n as follows:

$$\hat{f}_n = \lfloor \frac{1}{2}(f_{n-1} + f_{n-2}) \rfloor$$

$$e_n = f_n - \hat{f}_n \quad (6.14)$$



Lossless Predictive Coding

- Let's consider an explicit example. Suppose we wish to code the sequence $f_1; f_2; f_3; f_4; f_5 = 21, 22, 27, 25, 22$. For the purposes of the predictor, we'll invent an extra signal value f_0 , equal to $f_1 = 21$, and first transmit this initial value, uncoded:

$$\hat{f}_2 = 21, e_2 = 22 - 21 = 1;$$

$$\hat{f}_3 = \lfloor \frac{1}{2}(f_2 + f_1) \rfloor = \lfloor \frac{1}{2}(22 + 21) \rfloor = 21, \\ e_3 = 27 - 21 = 6;$$

$$\hat{f}_4 = \lfloor \frac{1}{2}(f_3 + f_2) \rfloor = \lfloor \frac{1}{2}(27 + 22) \rfloor = 24, \\ e_4 = 25 - 24 = 1;$$

$$\hat{f}_5 = \lfloor \frac{1}{2}(f_4 + f_3) \rfloor = \lfloor \frac{1}{2}(25 + 27) \rfloor = 26,$$

$$e_5 = 22 - 26 = -4 \tag{6.15}$$



Lossless Predictive Coding

- The error does center around zero, we see, and coding (assigning bit-string codewords) will be efficient. Fig. 6.16 shows a typical schematic diagram used to encapsulate this type of system:

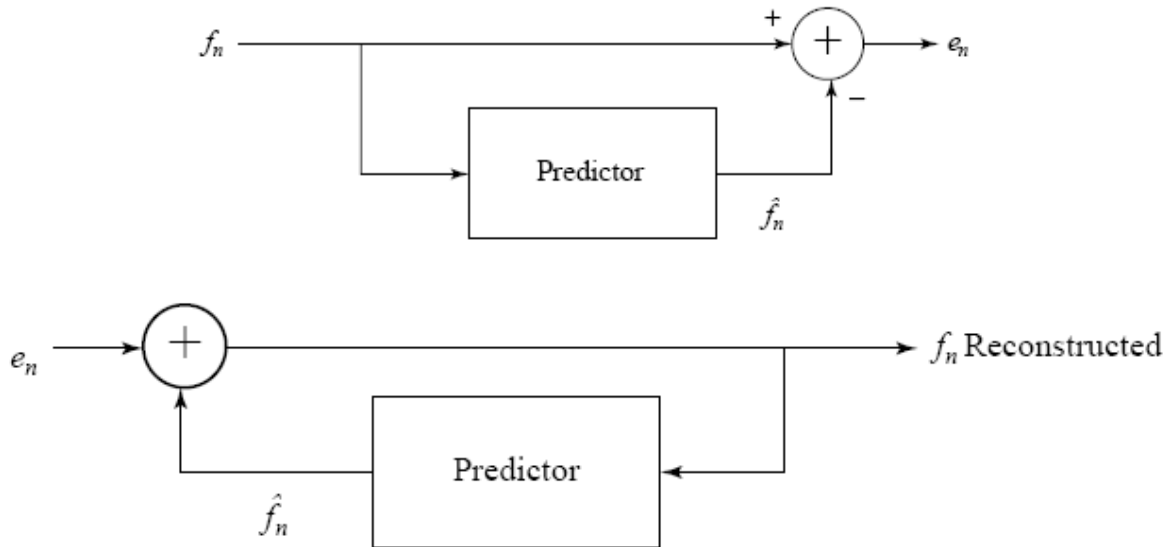


Fig. 6.16: Schematic diagram for Predictive Coding encoder and decoder.




DPCM

- ⦿ Differential PCM is exactly the same as Predictive Coding, except that it incorporates a *quantizer* step.
- ⦿ (a) One scheme for analytically determining the best set of quantizer steps, for a non-uniform quantizer, is the **Lloyd-Max** quantizer, which is based on a least-squares minimization of the error term.
- ⦿ (b) Our nomenclature: signal values: f_n -- the original signal, \hat{f}_n the predicted signal, and \tilde{f}_n the quantized, reconstructed signal.



DPCM

-  **DPCM:** form the prediction; form an error e_n by subtracting the prediction from the actual signal; then quantize the error to a quantized version, \tilde{e}_n . The set of equations that describe DPCM are as follows:

$$\hat{f}_n = \text{function_of}(\tilde{f}_{n-1}, \tilde{f}_{n-2}, \tilde{f}_{n-3}, \dots),$$

$$e_n = f_n - \hat{f}_n,$$

$$\tilde{e}_n = Q[e_n],$$

(6.16)

transmit *codeword*(\tilde{e}_n),

reconstruct: $\tilde{f}_n = \hat{f}_n + \tilde{e}_n$.

Then *codewords* for quantized error values \tilde{e}_n are produced using entropy coding, e.g. Huffman coding (Chapter 7).



Schematic diagram for DPCM

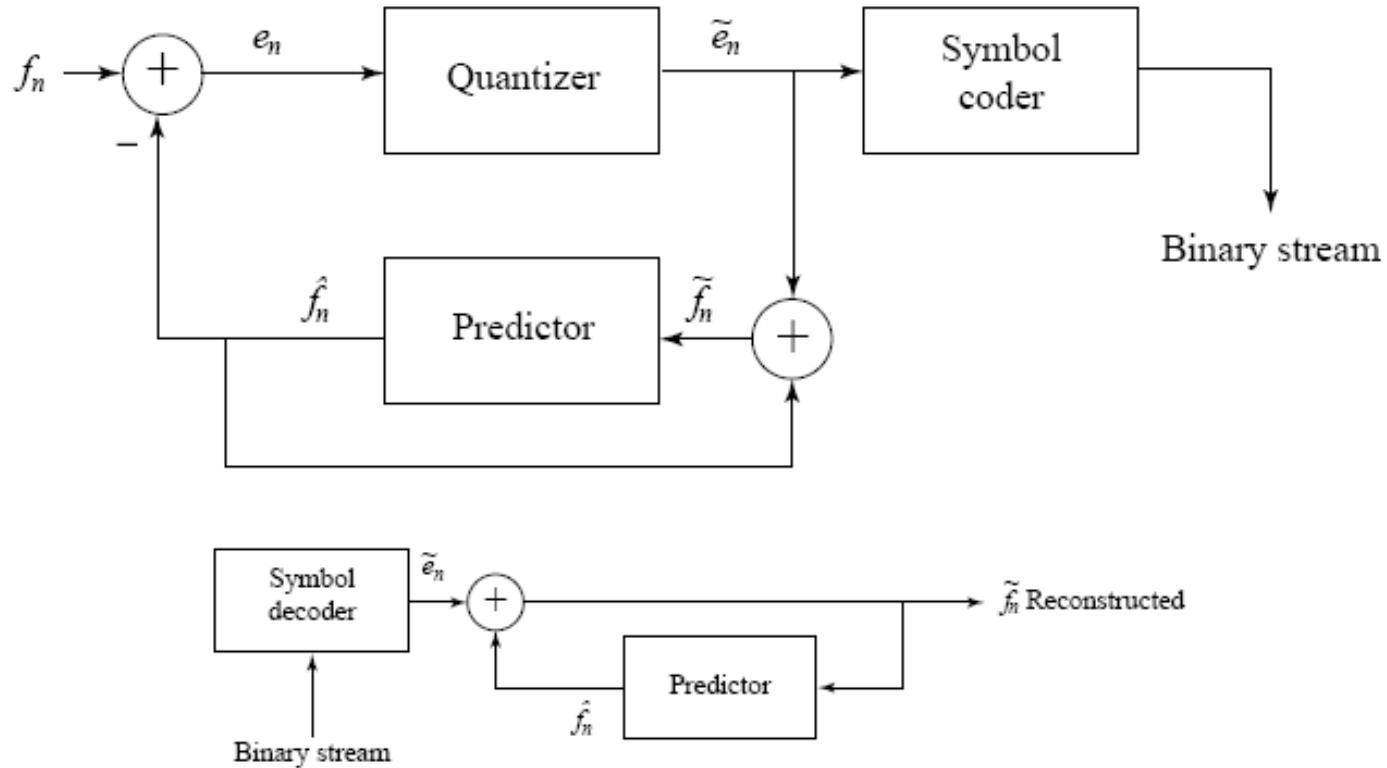


Fig. 6.17: Schematic diagram for DPCM encoder and decoder









6.4 Further Exploration



上海交通大學

Shanghai Jiao Tong University

<http://www.cs.sfu.ca/mmmbook/furtherv2/node6.html>

-  The Useful links included are:
-  An excellent discussion of the use of FM to create synthetic sound.
-  An extensive list of audio file formats.
-  CD audio file formats are somewhat different. The main music format is called “red book audio.” A good description of various CD formats is on the website.
-  A General MIDI Instrument Patch Map, along with a General MIDI Percussion Key Map.
-  A link to good tutorial on MIDI and wave table music synthesis.
-  A link to a java program for decoding MIDI streams.
-  A good multimedia/sound page, including a source for locating Internet sound/music materials.



上海交通大学

Shanghai Jiao Tong University
