V. Homomorphic Signal Processing

⊙ 5-A Homomorphism

Homomorphism is a way of "carrying over" operations from one algebra system into another.
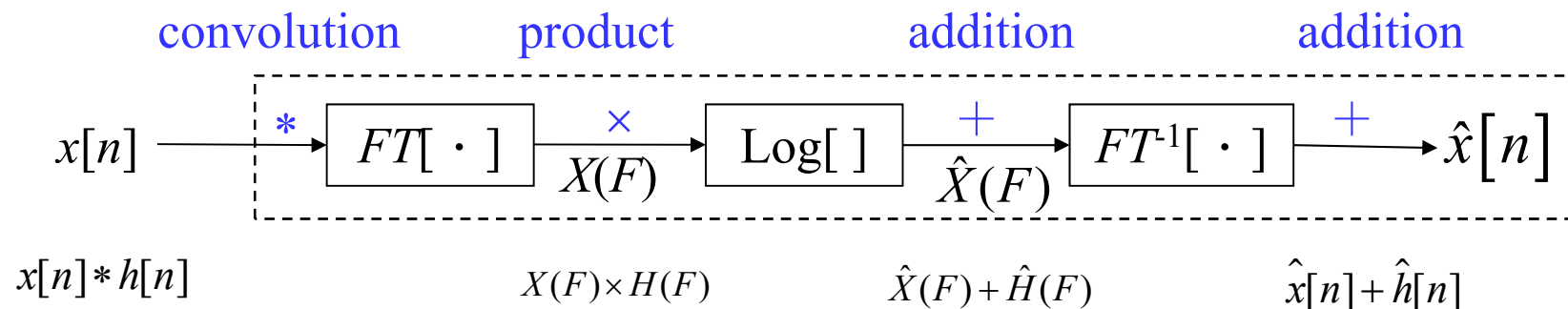
Ex.   convulution $\xrightarrow{\text{Fourier}}$ multiplication $\xrightarrow{\text{log}}$ addition
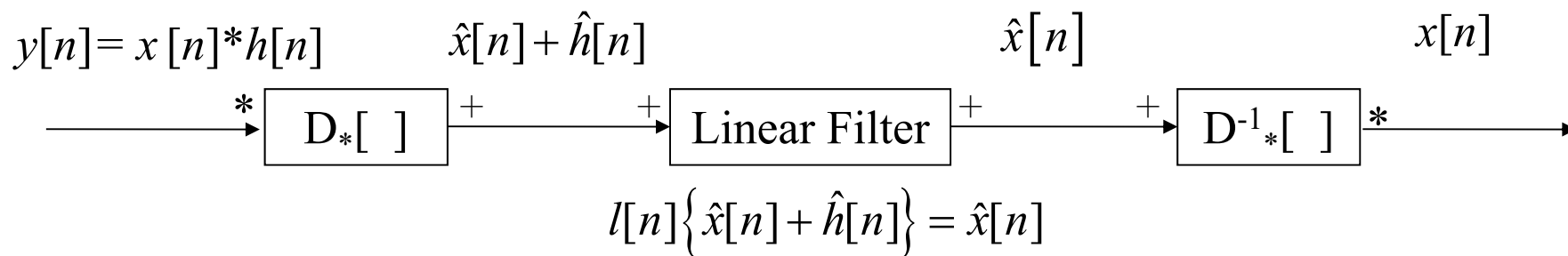
把複雜的運算，變成效能相同但較簡單的運算

⊙ 5-B Cepstrum

$$\hat{X}(Z)\Big|_{z=e^{i2\pi F}} = \log X(Z)\Big|_{z=e^{i2\pi F}} = \log\big|X(Z)\big|_{z=e^{i2\pi F}} + j\arg[X(e^{i2\pi F})]$$
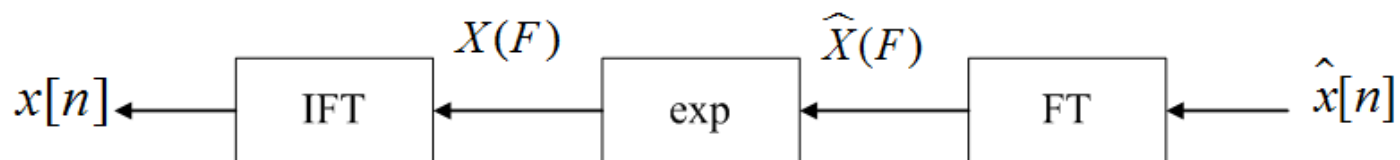
For the system $D_*[\ \cdot\ ]$

convolution   product   addition   addition

$$x[n] \longrightarrow \ *\ \boxed{FT[\ \cdot\ ]} \ \overset{\times}{\underset{X(F)}{\longrightarrow}} \ \boxed{\text{Log}[\ ]} \ \overset{+}{\underset{\hat{X}(F)}{\longrightarrow}} \ \boxed{FT^{-1}[\ \cdot\ ]} \ \overset{+}{\longrightarrow} \hat{x}[n]$$

$x[n]*h[n]$ \qquad\qquad $X(F)\times H(F)$ \qquad $\hat{X}(F)+\hat{H}(F)$ \qquad $\hat{x}[n]+\hat{h}[n]$

FT: discrete-time Fourier transform

- 由 $y[n]= x[n]*h[n]$ 重建 $x[n]$

$y[n]= x[n]*h[n]$ $\hat{x}[n]+\hat{h}[n]$ $\hat{x}[n]$ $x[n]$

$$* \quad \boxed{D_*[\ ]} \quad + \qquad + \quad \boxed{\text{Linear Filter}} \quad + \qquad + \quad \boxed{D^{-1}_*[\ ]} \quad *$$

$$l[n]\left\{\hat{x}[n]+\hat{h}[n]\right\} = \hat{x}[n]$$

For the system $D^{-1}_*[\ \cdot\ ]$

$$x[n] \leftarrow \boxed{\text{IFT}} \xleftarrow{X(F)} \boxed{\exp} \xleftarrow{\widehat{X}(F)} \boxed{\text{FT}} \leftarrow \hat{x}[n]$$

- 有趣的名詞

     $\hat{x}[n]$ cepstrum

     n quefrency

     $l[n]$ lifter

$\hat{x}[n]$ $\hat{h}[n]$

$\hat{l}[n]$
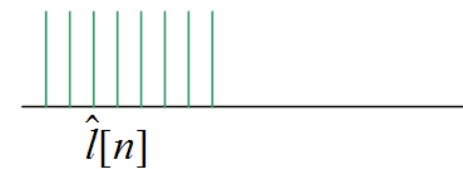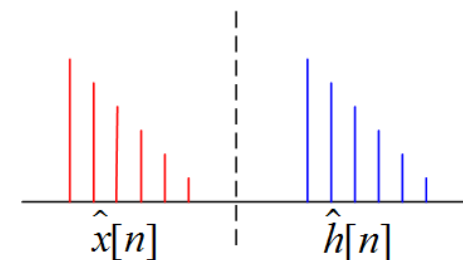
$$l[n]\left\{\hat{x}[n]+\hat{h}[n]\right\} = \hat{x}[n]$$

⦿ 5-C Complex Cepstrum and Real Cepstrum

- Complex Cepstrum

$$\hat{x}[n] = \int_{-1/2}^{1/2} \hat{X}(F) e^{i2\pi nF} dF \qquad : \text{inverse F.T}$$

$$\text{where} \quad \hat{X}(F) = \log|X(F)| + j\arg[X(F)]$$

ambiguity for phase

Problems: (1)

(2)

Actually, the COMPLEX Cepstrum is REAL for real input

- Real Cepstrum

$$C[n] = \int_{-1/2}^{1/2} \log|X(F)| e^{j2\pi Fn} dF$$

$$\Downarrow$$

$$REAL - valued$$

(but some problem of is not solved)

⊙ 5-D Methods for Computing the Cepstrum

• **Method 1**: Compute the inverse discrete time Fourier transform:

$$\hat{x}[n] = \int_{-1/2}^{1/2} \hat{X}(F)\, e^{i2\pi nF}\, dF$$

Problem: $\hat{X}(F)$ may be infinite

● Method 2 (From Poles and Zeros of the Z Transform) 實際上計算 cepstrum的方法

time delay

$$X(Z) = \frac{A Z^r \prod_{k=1}^{m_i}(1-a_k Z^{-1}) \prod_{k=1}^{m_0}(1-b_k Z)}{\prod_{k=1}^{P_i}(1-c_k Z^{-1}) \prod_{k=1}^{P_0}(1-d_k Z)}$$

where

$$\left| a_k \right|, \left| b_k \right|, \left| c_k \right|, \left| d_k \right| \leq 1$$

<u>Poles & zeros inside unite circle</u>   <u>Poles & zeros outside unite circle</u>

$$\therefore \hat{X}(Z) = \log X(Z) = \log A + r \cdot \log Z + \sum_{k=1}^{m_i} \log (1-a_k Z^{-1}) + \sum_{k=1}^{m_0} \log (1-b_k Z)$$

$$- \sum_{k=1}^{P_i} \log (1-c_k Z^{-1}) - \sum_{k=1}^{P_0} \log (1-d_k Z)$$

$$\therefore \hat{X}(Z) = \log X(Z) = \log A + r \cdot \log Z + \sum_{k=1}^{m_i} \log (1 - a_k Z^{-1}) + \sum_{k=1}^{m_0} \log (1 - b_k Z)$$

$$- \sum_{k=1}^{P_i} \log (1 - c_k Z^{-1}) - \sum_{k=1}^{P_0} \log (1 - d_k Z)$$

Z^{-1}

(inverse Z transform)

?

Taylor series

$$f(t) = f(t_0) + \sum_{n=1}^{\infty} \frac{f^{(n)}(t_0)}{n!} (t - t_0)^n$$

Taylor series expansion $\bigg| Z^{-1}$ (Suppose that $r = 0$)

$$\hat{x}[n] = \begin{cases} \log(A) & , n = 0 \\ -\sum\limits_{k=1}^{m_i} \dfrac{a_k{}^n}{n} + \sum\limits_{k=1}^{P_i} \dfrac{c_k{}^n}{n} & , n > 0 \\ \sum\limits_{k=1}^{m_0} \dfrac{b_k{}^{-n}}{n} - \sum\limits_{k=1}^{P_0} \dfrac{d_k{}^{-n}}{n} & , n < 0 \end{cases}$$

Poles & zeros inside unit circle, right-sided sequence

Poles & zeros outside unit circle, left-sided sequence

Note:

(1) 在 complex cepstrum domain
   Minimum phase 及 maximum phase 之貢獻以 $n = 0$ 為分界切開
(2) For FIR case, $c_k = 0$, $d_k = 0$
(3) The complex cepstrum is unique and of infinite duration for both positive
   & negative n, even though $x[n]$ is causal & of finite durations

$\hat{x}[n]$ is always IIR

● **Method 3**

$$Z \cdot \hat{X}'(Z) = Z \cdot \frac{X'(Z)}{X(Z)}$$

$$\therefore ZX'(Z) = Z\hat{X}'(Z) \cdot X(Z)$$

$$\Big\downarrow \ Z^{-1}$$

$$n\, x[n] = \sum_{k=-\infty}^{\infty} k\, \hat{x}[k]\, x[n-k]$$

$$\therefore x[n] = \sum_{k=-\infty}^{\infty} \frac{k}{n}\, \hat{x}[k]\, x[n-k] \qquad for\ n \neq 0$$

Suppose that $x[n]$ is <u>causal</u> and has <u>minimum phase</u>, i.e. $x[n] = \hat{x}[n] = 0$, $n < 0$

$$x[n] = \sum_{k=-\infty}^{\infty} \frac{k}{n}\hat{x}[k]x[n-k] \qquad \textit{for } n \neq 0$$

$$\Rightarrow x[n] = \sum_{k=0}^{n} \frac{k}{n}\hat{x}[k]x[n-k] \qquad \textit{for } n > 0 \qquad (\textit{causal sequence})$$

$$x[n] = \hat{x}[n]x[0] + \sum_{k=0}^{n-1} \frac{k}{n}\hat{x}[k]x[n-k]$$

For a minimum phase sequence $x[n]$

$$\hat{x}[n] = \begin{cases} 0 & ,n < 0 \\ \dfrac{x[n]}{x[0]} - \displaystyle\sum_{k=0}^{n-1}(\frac{k}{n})\hat{x}[k]\dfrac{x[n-k]}{x[0]} & ,n > 0 \qquad \text{recursive method} \\ \log A & ,n = 0 \end{cases}$$

For anti-causal and maximum phase sequence, $x[n] = \hat{x}[n] = 0, \, n > 0$

$$x[n] = \sum_{k=n}^{0} \frac{k}{n} \hat{x}[k] x[n-k] \qquad , n < 0$$

$$= \hat{x}[n] x[0] + \sum_{k=n+1}^{0} \frac{k}{n} \hat{x}[k] x[n-k]$$

For maximum phase sequence,

$$\hat{x}[n] = \begin{cases} 0 & , n > 0 \\ \log A & , n = 0 \\ \dfrac{x[n]}{x[0]} - \sum_{k=n+1}^{0} (\dfrac{k}{n}) \hat{x}[k] \dfrac{x[n-k]}{x[0]} & , n < 0 \end{cases}$$

⊙ 5-E Properties

P.1) The complex cepstrum decays at least as fast as $\dfrac{1}{n}$

$$\left| \hat{x}[n] \right| < c \left| \frac{\alpha^n}{n} \right| \qquad -\infty < n < \infty$$

$$\alpha = \max\left(a_k, b_k, c_k, d_k \right)$$

P.2) If $X(Z)$ has no poles and zeros outside the unit circle, i.e. $x[n]$ is minimum phase, then

$$\hat{x}[n] = 0 \qquad \text{for all } n < 0$$

because of no b_k, d_k

P.3) If $X(Z)$ has no poles and zeros inside the unit circle, i.e. $x[n]$ is maximum phase, then
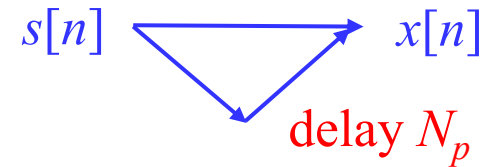
$$\hat{x}[n] = 0 \qquad \text{for all } n > 0$$

because of no a_k, c_k

P.4) If $x[n]$ is of finite duration, then

$$\hat{x}[n] \text{ has infinite duration}$$

◎ 5-F Application of Homomorphic Deconvolution

(1) Equalization for Echo

$$x[n] = s[n] + \alpha \, s[n - N_p]$$

$s[n]$ $x[n]$

delay N_p

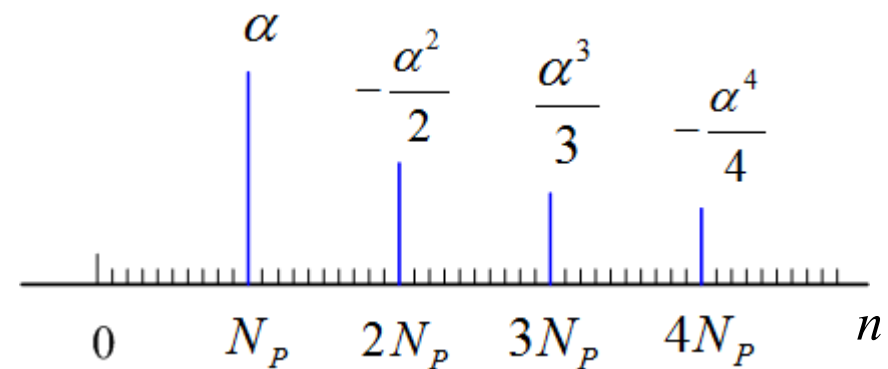Let $p[n]$ be $p[n] = \delta[n] + \alpha \delta[n - N_p]$

$$x[n] = s[n] + \alpha \, s[n - N_p] = s[n] * p[n]$$
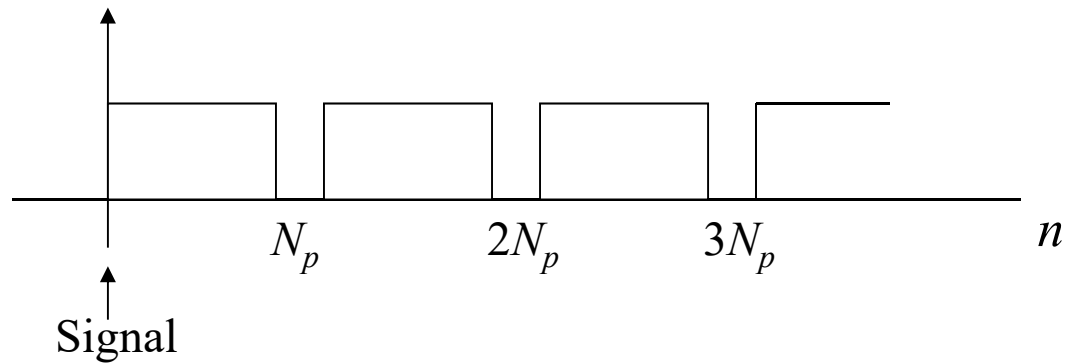
$$P(Z) = 1 + \alpha Z^{-N_p}$$

$$\hat{P}(Z) = \log(1 + \alpha Z^{-N_p}) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{\alpha^k}{k} Z^{-kN_p}$$

$\downarrow Z^{-1}$

$$\hat{p}[n] = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{\alpha^k}{k} \delta(n - k \cdot N_p)$$

α $-\dfrac{\alpha^2}{2}$ $\dfrac{\alpha^3}{3}$ $-\dfrac{\alpha^4}{4}$

0 N_P $2N_P$ $3N_P$ $4N_P$ n

Filtering out the echo by the following "lifter":

(2) Representation of acoustic engineering

$$x[n] \quad = \quad s[n] \quad * \quad h[n]$$

Synthesiz   music   building effect：e.g. 羅馬大教堂的
ed music   impulse response

(3) Speech analysis

$$s[n] \quad = \quad g[n] \quad * \quad v[n] \quad * \quad p[n]$$

<u>Speech wave</u>   <u>Global wave shape</u>   <u>Vocal tract impulse</u>   <u>Pitch</u>

They can be separated by filtering in the complex Cepstrum Domain

(4) Seismic Signals

(5) Multiple-path problem analysis

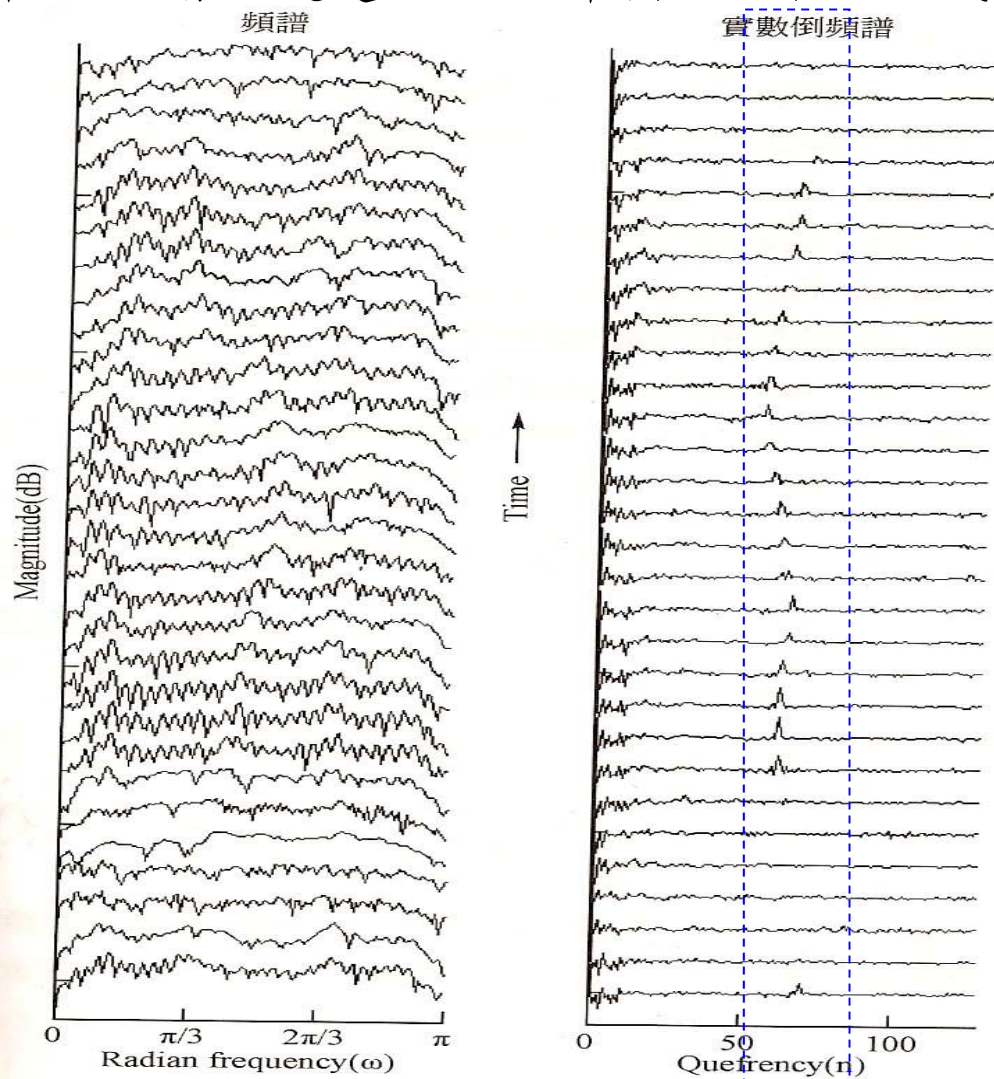用 cepstrum 將 multipath 的影響去除

From 王小川，"語音訊號處理"，全華出版，台北，民國94年。

圖 7-5 元音頻譜與倒頻譜
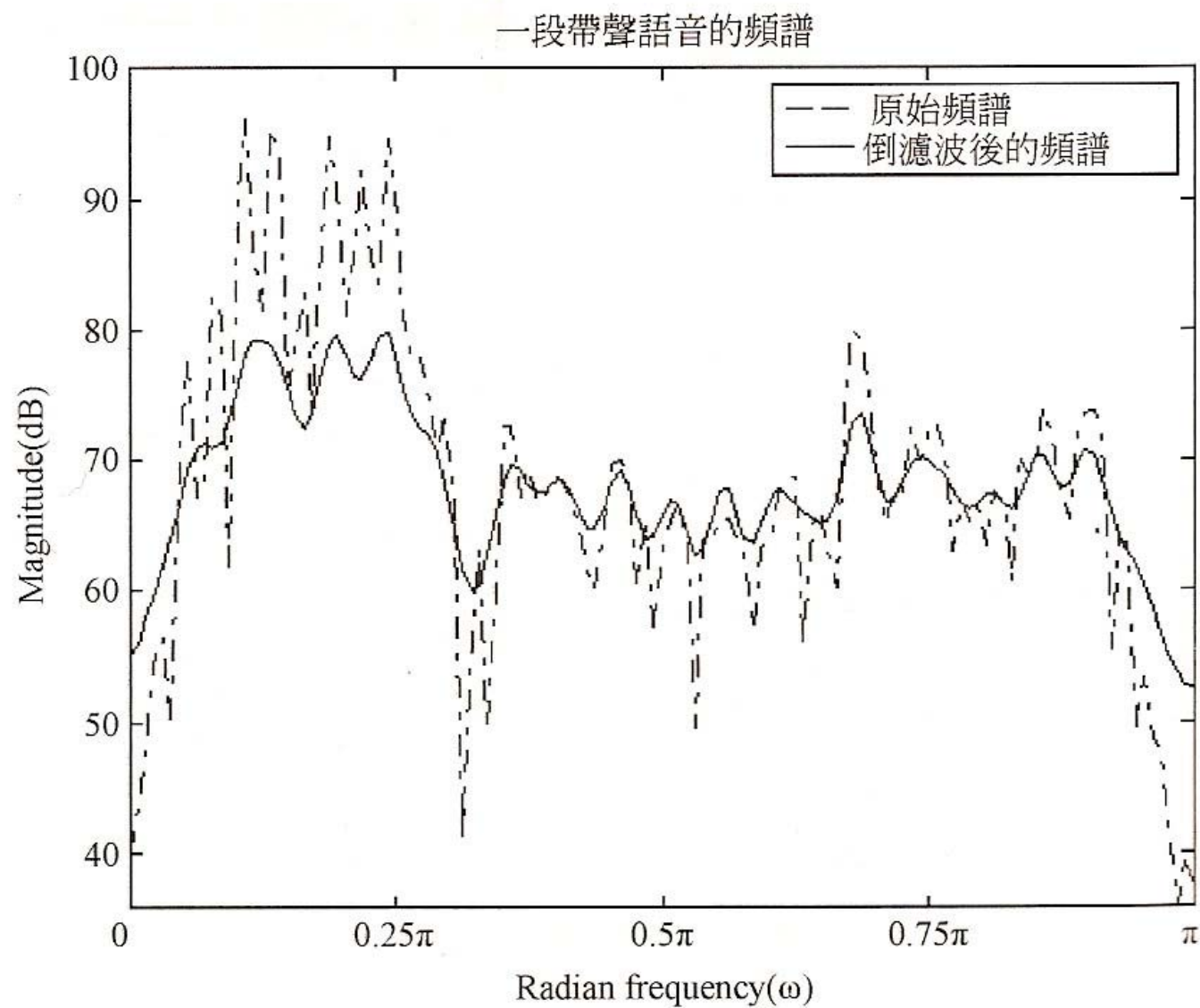
From 王小川，"語音訊號處理"，全華出版，台北，民國94年。

一段帶聲語音的頻譜

圖 7-6　經過倒濾波器作平滑處理的頻譜

⊙ 5-G Problems of Cepstrum

(1) $|\log(X(Z))|$

(2) Phase

(3) Delay Z^{-k}

(4) Only suitable for the multiple-path-like problem

⊙ 5-H Differential Cepstrum

$$\hat{x}_d(n) = Z^{-1}\left[\frac{X'(Z)}{X(Z)}\right] \qquad \text{或} \qquad \hat{x}_d[n] = \int_{-1/2}^{1/2} \frac{X'(F)}{X(F)} e^{i2\pi F} dF$$

inverse Z transform

Note: $\dfrac{d}{dZ}\hat{X}(Z) = \dfrac{d}{dZ}\log(X(Z)) = \dfrac{X'(Z)}{X(Z)}$

If $\quad x(n) = x_1(n) * x_2(n)$

$X(Z) = X_1(Z) \cdot X_2(Z)$

$X'(Z) = X_1'(Z) \cdot X_2(Z) + X_1(Z) \cdot X_2'(Z)$

$\dfrac{X'(Z)}{X(Z)} = \dfrac{X_1'(Z)}{X_1(Z)} + \dfrac{X_2'(Z)}{X_2(Z)} \qquad \therefore \quad \hat{x}_d(n) = \hat{x}_{1d}(n) + \hat{x}_{2d}(n)$

Advantages: no phase ambiguity

able to deal with the delay problem

● **Properties of Differential Cepstrum**

(1) The differential Cepstrum is shift & scaling invariant

不只適用於 multi-path-like problem

也適用於 pattern recognition

If $y[n] = A \, X[n - r]$

$$\Rightarrow \quad \hat{y}_d(n) = \begin{cases} \hat{x}_d(n) & , \quad n \neq 1 \\ -r + \hat{x}_d(1) & , \quad n = 1 \end{cases}$$

(Proof): $Y(z) = A z^{-r} X(z)$

$$Y'(z) = A z^{-r} X'(z) - r A z^{-r-1} X(z)$$

$$\frac{Y'(z)}{Y(z)} = \frac{X'(z)}{X(z)} - r z^{-1}$$

(2) The complex cepstrum $\hat{C}[n]$ is closely related to its differential cepstrum $\hat{x}_d[n]$ and the signal original sequence $x[n]$

$$\hat{C}(n) = \frac{-\hat{x}_d(n+1)}{n} \qquad n \neq 0 \qquad \textit{diff cepstrum}$$

$$\textit{and} \quad -(n-1)\,x(n-1) = \sum_{k=-\infty}^{\infty} \hat{x}_d(n)\,x(n-k) \qquad \textit{recursive formula}$$

Complex cepstrum 做得到的事情, differential cepstrum 也做得到！

(3) If $x[n]$ is minimum phase (no poles & zeros outside the unit circle), then
$\hat{x}_d[n] = 0$ for $n \le 0$

(4) If $x[n]$ is maximum phase (no poles & zeros inside the unit circle), then
$\hat{x}_d[n] = 0$ for $n \ge 2$

delay

max phase   ↙   min phase

$$\cancel{}\ \Big|\ \ \ \Big|\ \ \ \Big|\ \cancel{}$$

0   1   2

(5) If $x(n)$ is of finite duration, $\hat{x}_d[n]$ has infinite duration

Complex cepstrum decay rate $\propto \dfrac{1}{n}$

Differential Cepstrum decay rate 變慢了, $\because$ $\hat{x}_d(n+1) = n \cdot \hat{c}(n) \propto n \cdot \dfrac{1}{n} = 1$

⊙ 5-I Mel-Frequency Cepstrum (梅爾頻率倒頻譜)

Take log in the frequency mask

gain　　　　　　　　　　　mask of Mel-frequency cepstrum

$f_1, f_2, f_3 \ldots\ldots$ 以等比級數增加

frequency
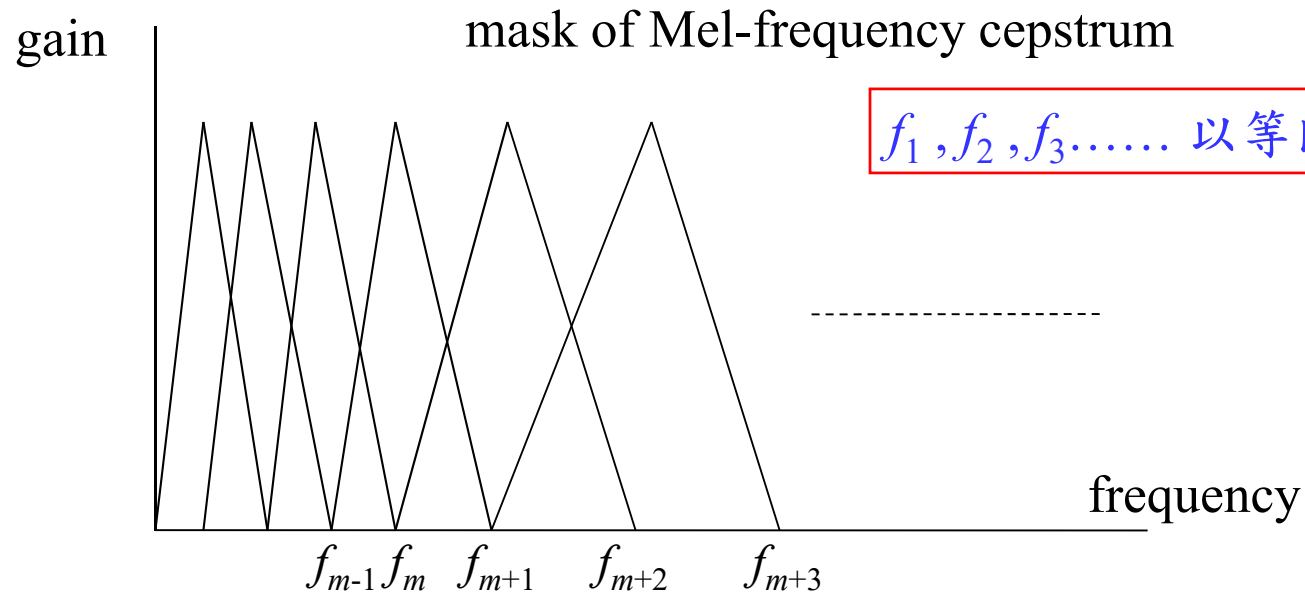
$f_{m-1} f_m \quad f_{m+1} \quad f_{m+2} \quad f_{m+3}$

$B_m[k] = 0$　　　　　　　　　　for $f < f_{m-1}$ and $f > f_{m+1}$

$B_m[k] = (k - f_{m-1})/(f_m - f_{m-1})$　　for $f_{m-1} \leq f \leq f_m$

$B_m[k] = (f_{m+1} - k)/(f_{m+1} - f_m)$　　for $f_m \leq f \leq f_{m+1}$

$f = k * f_s / N$

Process of the Mel-Cepstrum

(1) $x[n] \xrightarrow{FT} X[k]$

(2) $Y[m] = \log \left\{ \sum_{k=f_{m-1}}^{f_{m+1}} |X[k]|^2 B_m[k] \right\}$

(3) $c_x[n] = \dfrac{1}{M} \sum_{m=1}^{M} Y[m] \cos\left(\dfrac{\pi n(m-1/2)}{M} \right)$

Q: What are the difference between the Mel-cepstrum and the original cepstrum?

Advantages :

Mel-frequency cepstrum 更接近人耳對語音的區別性

用 $c_x[1], c_x[2], c_x[3], \ldots\ldots, c_x[13]$ 即足以描述語音特徵

◉ 5-J References

- R. B. Randall and J. Hee, "Cepstrum analysis," *Wireless World*, vol. 88, pp. 77-80. Feb. 1982

- 王小川，"語音訊號處理"，全華出版，台北，民國94年。

- A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, London: Prentice-Hall, 3rd ed., 2010.

- S. C. Pei and S. T. Lu, "Design of minimum phase and FIR digital filters by differential cepstrum," *IEEE Trans. Circuits Syst. I*, vol. 33, no. 5, pp. 570-576, May 1986.

- S. Imai, "Cepstrum analysis synthesis on the Mel-frequency scale," *ICASSP*, vol. 8, pp. 93-96, Apr. 1983.

附錄六：聲音檔和影像檔的處理 (by Matlab)

A. 讀取聲音檔

- 電腦中，沒有經過壓縮的聲音檔都是 *.wav 的型態

- 讀取： **wavread**

    註：2015版本以後的 Matlab，**wavread** 將改為 **audioread**

- 例： [**x**, **fs**] = wavread('C:\WINDOWS\Media\ringin.wav');
  可以將 ringin.wav 以數字向量 **x** 來呈現。 **fs**: sampling frequency
  這個例子當中 size(**x**) = 9981 1   fs = 11025

- 思考: 所以，取樣間隔多大?

- 這個聲音檔有多少秒？

畫出聲音的波型

　　　time = [0:length(x)-1]/fs;　　% x 是前頁用 wavread 所讀出的向量

plot(time, x)

鈴聲的波型
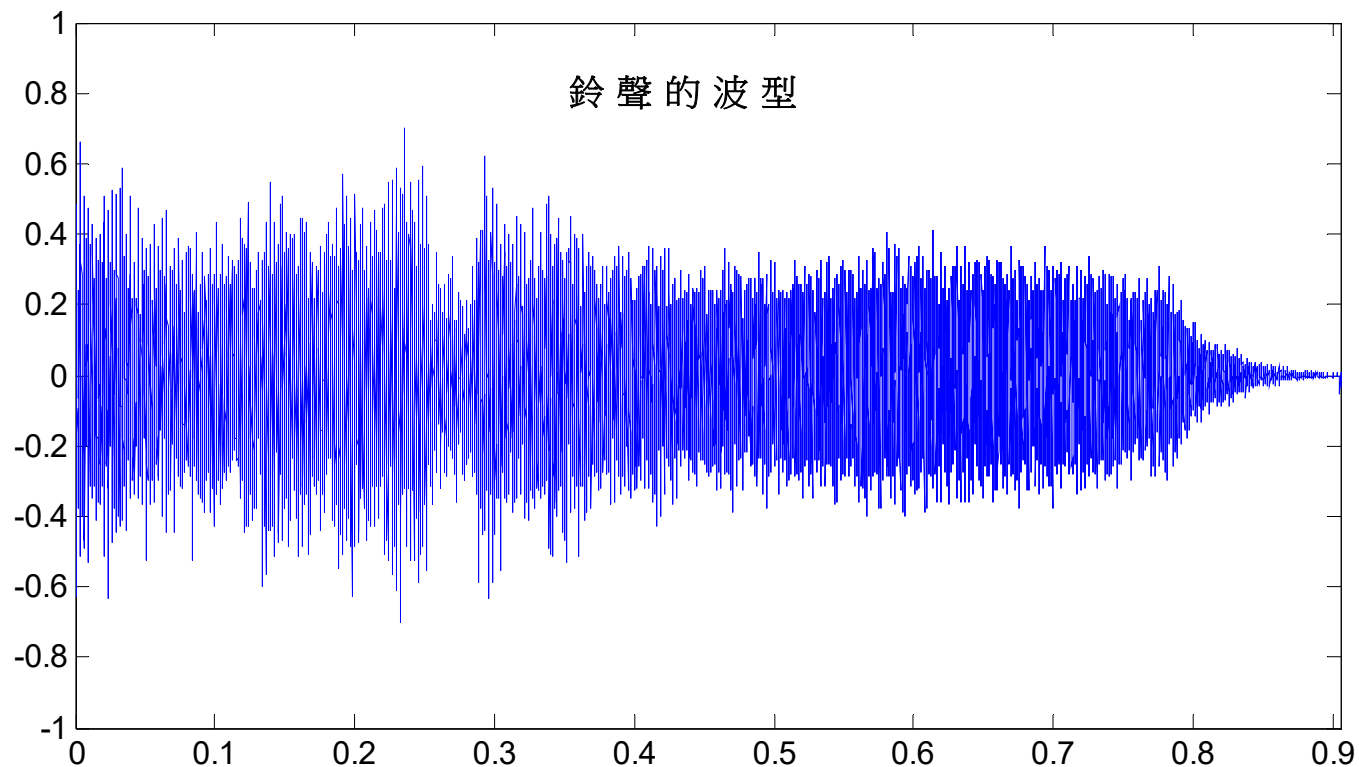
注意：*.wav 檔中所讀取的資料，值都在 −1 和 +1 之間

一個聲音檔如果太大，我們也可以只讀取它部分的點

[x, fs]=wavread('C:\WINDOWS\Media\ringin.wav', [4001 5000]);

% 讀取第4001至5000點

[x, fs, **nbits**] = wavread('C:\WINDOWS\Media\ringin.wav');

nbits: x(n) 的bit 數
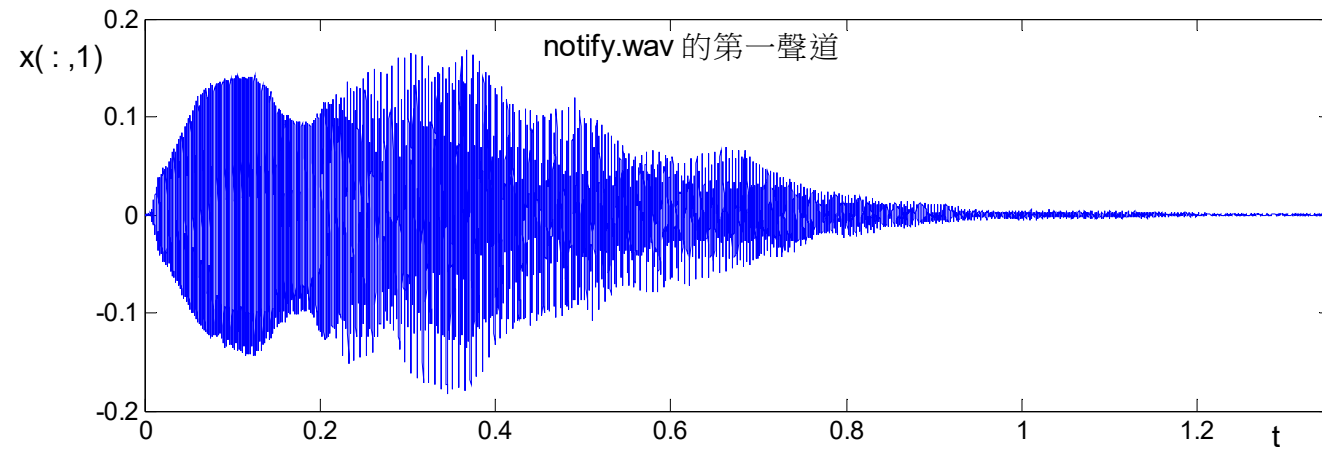
第一個bit : 正負號，第二個bit : 2^{-1}，第三個bit : 2^{-2}， ….. ，

第 n 個bit : $2^{-\text{nbits} +1}$， 所以 **x** 乘上$2^{\text{nbits} -1}$ 是一個整數

以鈴聲的例子， nbits = 8，所以 **x** 乘上 128是個整數

●有些聲音檔是 雙聲道（**Stereo**）的型態 (俗稱立體聲)

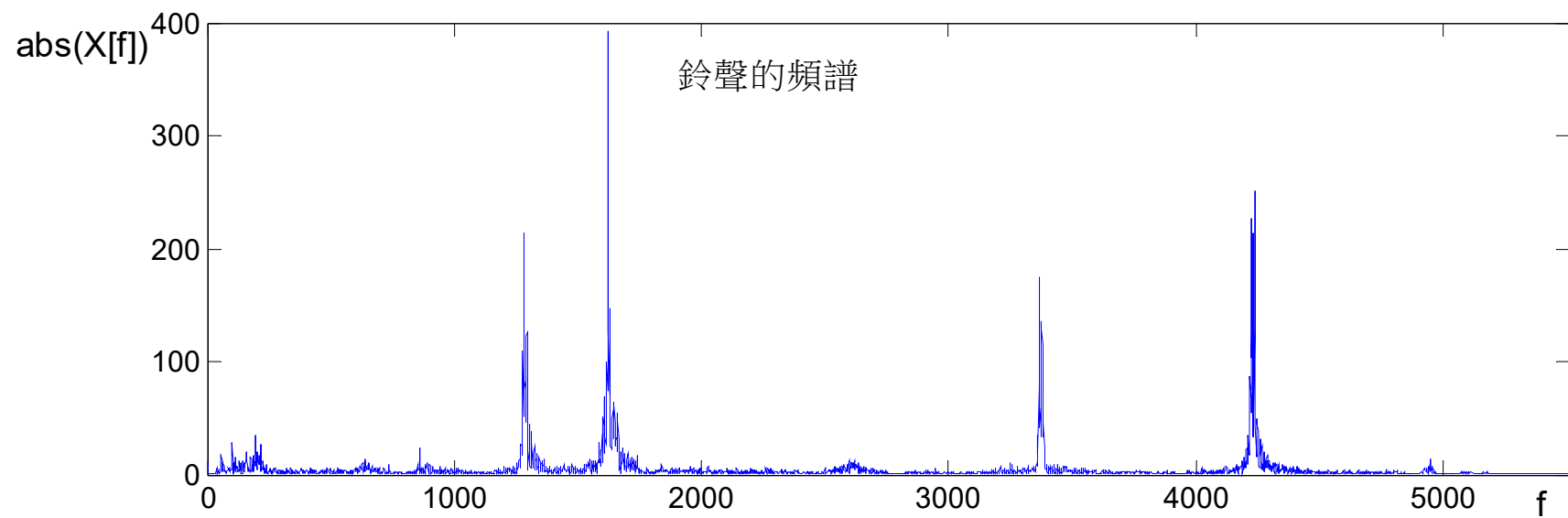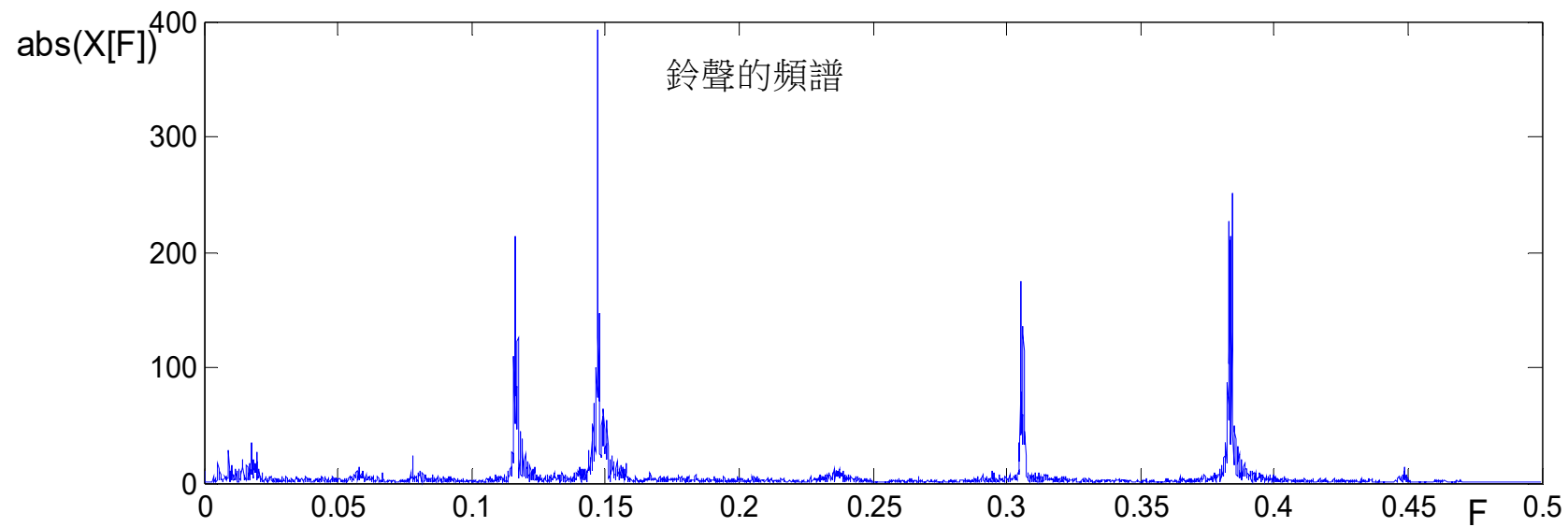例： [x, fs]=wavread('C:\WINDOWS\Media\notify.wav');

size(x) = 29823　2　　　　fs = 22050
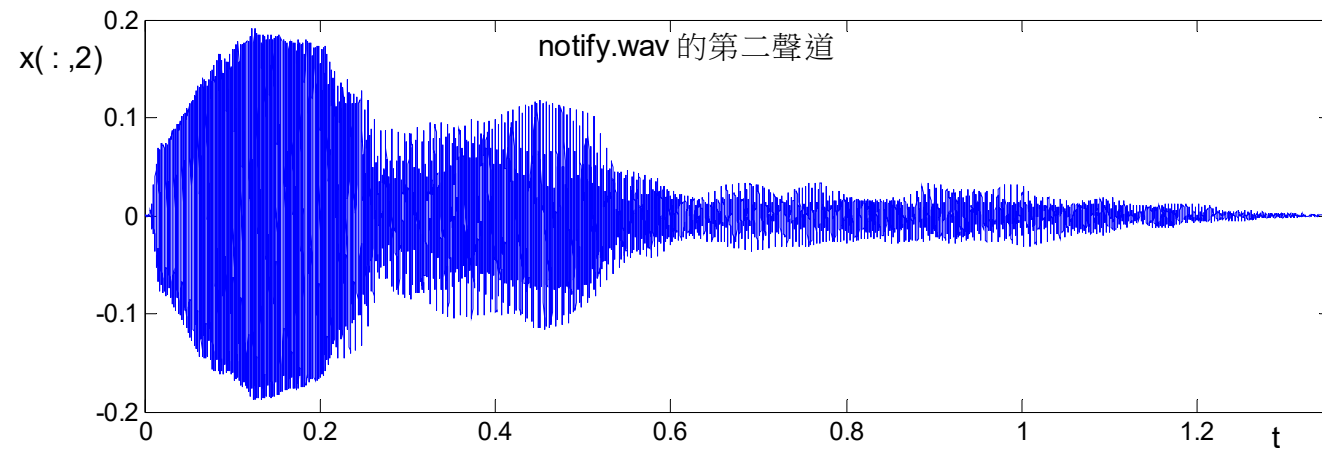
B. 繪出頻譜 (請參考附錄四)

X = fft(x);   plot(abs(X))

鈴聲的頻譜
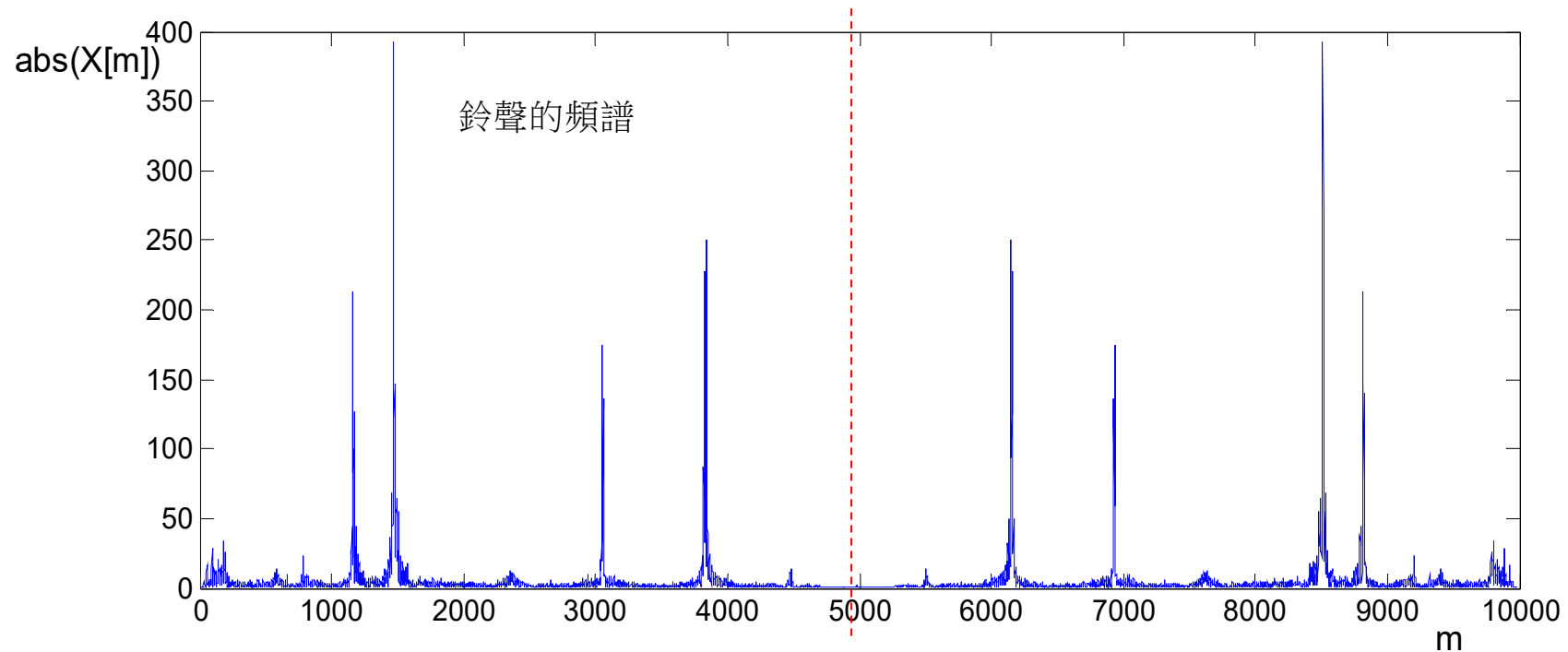
fft 橫軸 轉換的方法
   (1) Using normalized frequency F: $F = m / N$.
   (2) Using frequency f, $f = F \times f_s = m \times (f_s / N)$.

鈴聲的頻譜

鈴聲的頻譜

C. 聲音的播放

(1) wavplay(**x**): 將 **x** 以 11025Hz 的頻率播放

(時間間隔 = 1/11025 = 9.07×10^{-5} 秒)

(2) sound(**x**): 將 **x** 以 8192Hz 的頻率播放

(3) wavplay(**x, fs**) 或 sound(**x, fs**): 將 **x** 以 **fs** Hz 的頻率播放

  Note: (1)~(3) 中 **x** 必需是1 個column (或2個 columns)，且 **x** 的值應該 介於 −1 和 +1 之間

(4) soundsc(**x, fs**): 自動把 **x** 的值調到 −1 和 +1 之間 再播放

D. 用 Matlab 製作 *.wav 檔：wavwrite

wavwrite(x, fs, waveFile)

　　　　將數據 **x** 變成一個 *.wav 檔，取樣速率為 **fs** Hz

① **x** 必需是1 個column (或2個 columns)　② **x** 值應該 介於 –1 和 +1 之間

③ 若沒有設定fs，則預設的fs 為 8000Hz

E. 用 Matlab 錄音的方法

錄音之前，要先將電腦接上麥克風，且確定電腦有音效卡

(部分的 notebooks 不需裝麥克風即可錄音)

範例程式：

```
Sec = 3;
Fs = 8000;
recorder = audiorecorder(Fs, 16, 1);
recordblocking(recorder, Sec);
audioarray = getaudiodata(recorder);
```

執行以上的程式，即可錄音。

錄音的時間為三秒，sampling frequency 為 8000 Hz

錄音結果為 audioarray，是一個 column vector (如果是雙聲道，則是兩個 column vectors)

範例程式 (續)：

```
wavplay(audioarray, Fs);                  % 播放錄音的結果

t = [0:length(audioarray)-1]./Fs;

plot (t, audioarray');                    % 將錄音的結果用圖畫出來

xlabel('sec','FontSize',16);

wavwrite(audioarray, Fs, 'test.wav')   % 將錄音的結果存成 *.wav 檔
```

指令說明：

recorder = audiorecorder(Fs, nb, nch);　　(提供錄音相關的參數)

　　Fs: sampling frequency,

　　nb: using nb bits to record each data

　　nch: number of channels (1 or 2)

recordblocking(recorder, Sec);　　(錄音的指令)

　　recorder: the parameters obtained by the command "audiorecorder"

　　Sec: the time length for recording

audioarray = getaudiodata(recorder);

　　(將錄音的結果，變成 audioarray 這個 column vector，如果是雙聲道，則 audioarray 是兩個 column vectors)

以上這三個指令，要並用，才可以錄音

F、MP3 檔的讀和寫

要先去這個網站下載 mp3read.m, mp3write.m 的程式

http://www.mathworks.com/matlabcentral/fileexchange/13852-mp3read-and-mp3write
程式原作者：Dan Ellis

mp3read.m ： 讀取 mp3 的檔案
mp3write.m ： 製作 mp3 的檔案

不同於 *.wav 檔 (未壓縮過的聲音檔)，*.mp3 是經過 MPEG-2 Audio Layer III的技術壓縮過的聲音檔

範例：

%% Write an MP3 file by Matlab

fs=8000;   % sampling frequency
t = [1:fs*3]/3;
filename = 'test';
Nbit=32;   % number of bits per sample
x= 0.2*cos(2*pi*(500*t+300*(t-1.5).^3));
mp3write(x, fs, Nbit, filename);   % make an MP3 file test.mp3

%% Read an MP3 file by Matlab

[x1, fs1]=mp3read('phase33.mp3');
x2=x1(577:end);   % delete the head
sound(x2, fs1)

F：影像檔的處理

Image 檔讀取: imread

Image 檔顯示: imshow, image, imagesc

Image 檔製作: imwrite

基本概念：灰階影像在 Matlab 當中是一個矩陣

彩色影像在 Matlab 當中是三個矩陣，分別代表 Red, Green, Blue

*.bmp: 沒有經過任何壓縮處理的圖檔

*.jpg: 有經過 JPEG 壓縮的圖檔

Video 檔讀取: aviread

範例一：(黑白影像)

im=imread('C:\Program Files\MATLAB\pic\Pepper.bmp');
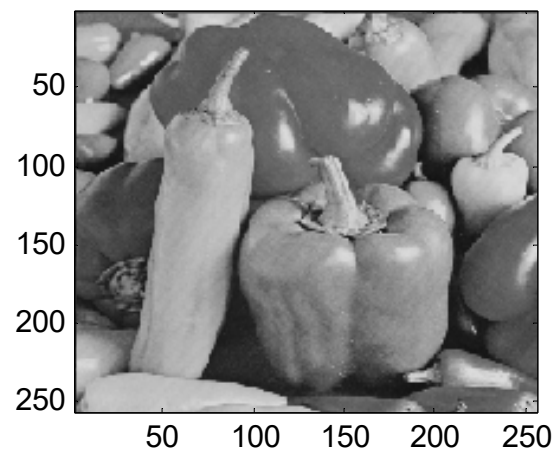
(注意，如果 Pepper.bmp 是個灰階圖，im 將是一個矩陣)

size(im)　　　(用 size 這個指令來看 im 這個矩陣的大小)

ans =

　256　256

image(im);
colormap(gray(256))

範例二：(彩色影像)

im2=imread('C:\Program Files\MATLAB\pic\Pepper512c.bmp');

size(im2)

ans =

　512　512　3

(注意，由於這個圖檔是個彩色的，所以 im2 將由三個矩陣複合而成)

imshow(im);

注意：要對影像做運算時，<u>要先變成 double 的格式</u>

否則電腦會預設影像為 integer 的格式，在做浮點運算時會產生誤差

例如，若要對影像做 2D Discrete Fourier transform

```
im=imread('C:\Program Files\MATLAB\pic\Pepper.bmp');
im=double(im);
Imf=fft2(im);
```