V. Homomorphic Signal Processing

◉ 5-A Homomorphism

Homomorphism is a way of "carrying over" operations from one algebra system into another.

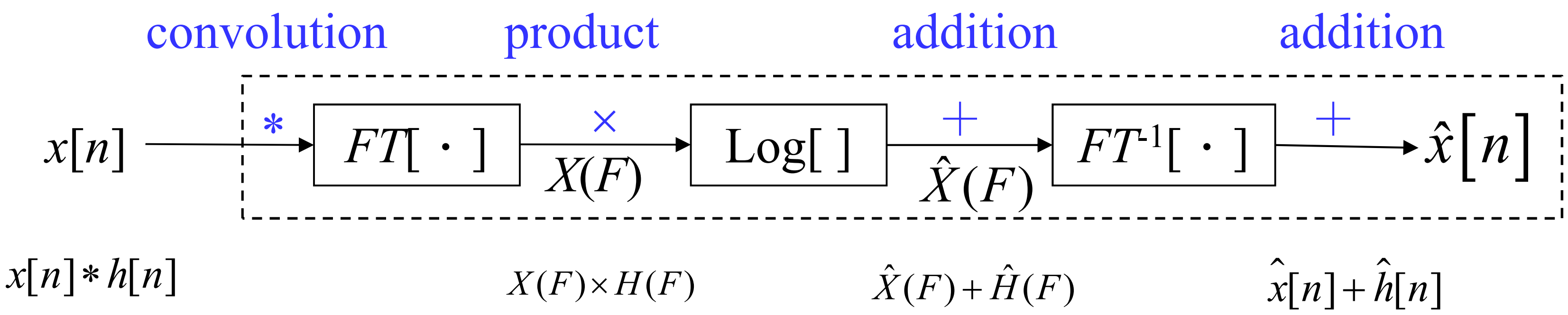
Ex. convulution $\xrightarrow[\textit{Fourier}]{}$ multiplication $\xrightarrow[\log]{}$ addition

把複雜的運算，變成效能相同但較簡單的運算

◉ 5-B Cepstrum

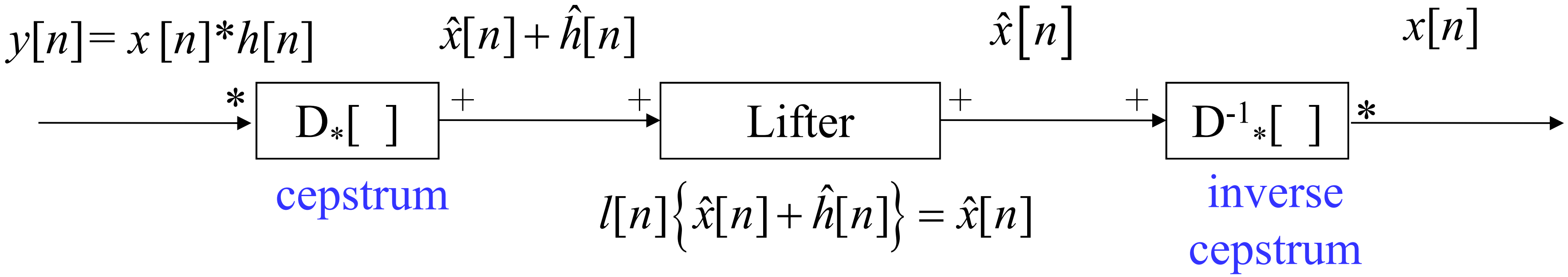
$$\hat{X}(Z)\Big|_{z=e^{i2\pi F}} = \log X(Z)\Big|_{z=e^{i2\pi F}} = \log\left|X(Z)\right|_{z=e^{i2\pi F}} + j\arg[X(e^{i2\pi F})]$$

For the process of cepstrum (denoted by $D_*[\cdot]$)

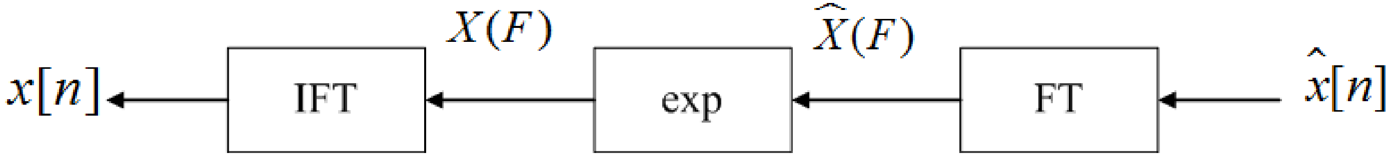


FT: discrete-time Fourier transform

- 由 $y[n]=x[n]*h[n]$ 重建 $x[n]$

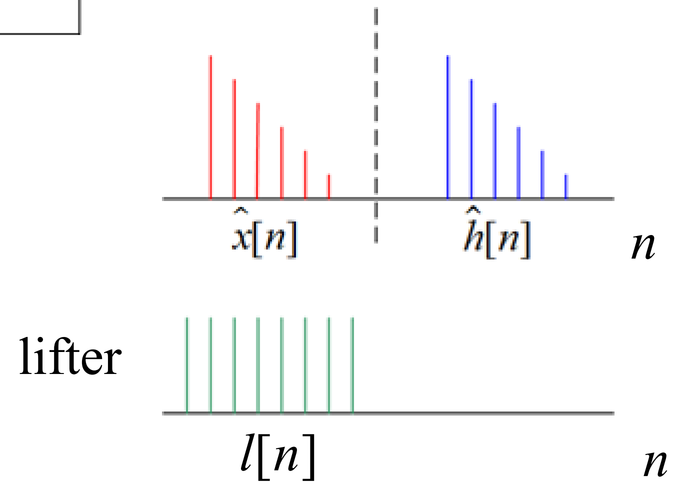


For the process of the inverse cepstrum $D^{-1}_*[\cdot]$



- 有趣的名詞

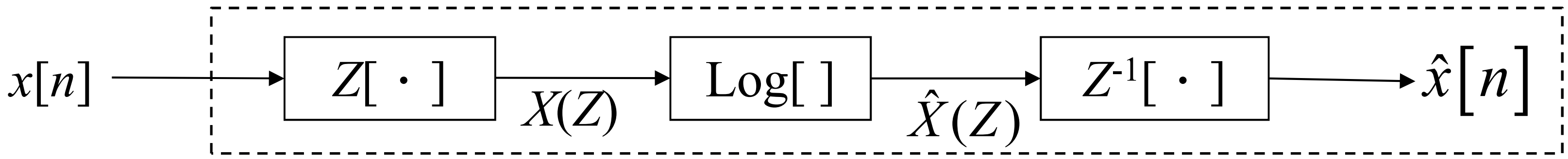
$\hat{x}[n]$	cepstrum
n	quefrency
$l[n]$	lifter



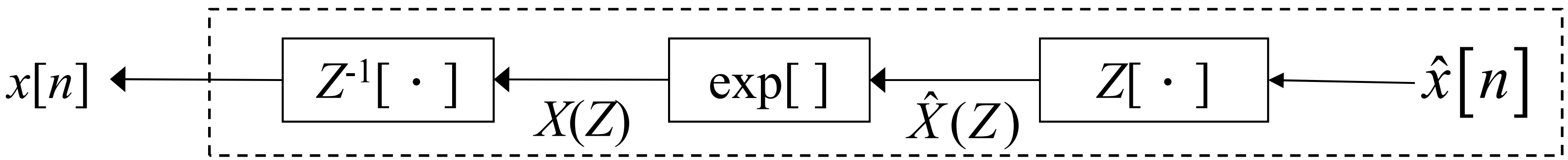
$$l[n]\left\{\hat{x}[n]+\hat{h}[n]\right\}=\hat{x}[n]$$

Using the Z transforms instead of the Fourier transforms:

For the process of cepstrum



For the process of the inverse cepstrum



◉ 5-C Methods for Computing the Cepstrum

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

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

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

• Method 2 (From Poles and Zeros of the Z Transform)

實際上計算 cepstrum的方法

time delay

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

where

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

a_k: zeros inside unit circle

c_k: poles inside unit circle

b_k^{-1}: zeros outside unit circle

d_k^{-1}: poles outside unit circle

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

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

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

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

$$\Big\downarrow Z^{-1} \text{ (inverse Z transform)}$$

?

Taylor series

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

$\boxed{\text{Taylor series expansion}}$ $\Big\downarrow Z^{-1}$ (Suppose that $r = 0$)

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

Poles & zeros inside unit circle, right-sided sequence

Poles & zeros outside unit circle, left-sided sequence

Note:

(1) $\hat{x}[n]$ always decays with $|n|$.

(2) 在 complex cepstrum domain

Minimum phase 及 maximum phase 之貢獻以 $n = 0$ 為分界切開

(3) For FIR case, there is no c_k and d_k

(4) The complex cepstrum is unique and of infinite duration for both positive & negative n, even though $x[n]$ is causal & of finite durations

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

- **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 causal and has minimum phase, i.e. $x[n] = \hat{x}[n] = 0,\ n < 0$

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

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

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

For a minimum phase sequence $x[n]$

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

Determining $\hat{x}[n]$ from $\hat{x}[0], \hat{x}[1], \cdots, \hat{x}[n-1]$

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

$$
\begin{aligned}
x[n] &= \sum_{k=n}^{0} \frac{k}{n}\hat{x}[k]x[n-k] \qquad\qquad ,n<0 \\
&= \hat{x}[n]x[0] + \sum_{k=n+1}^{0} \frac{k}{n}\hat{x}[k]x[n-k]
\end{aligned}
$$

For maximum phase sequence,

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

◉ 5-D Properties

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

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

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

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

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

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

$$\hat{x}[n] = 0 \qquad \textit{for all } n > 0$$

because of no a_k, c_k

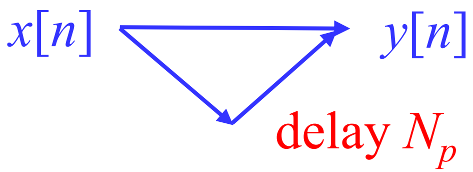
P.4) If $x[n]$ is of finite duration, then

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

◉ 5-E Application of Homomorphic Deconvolution

(1) Equalization for Echo

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



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

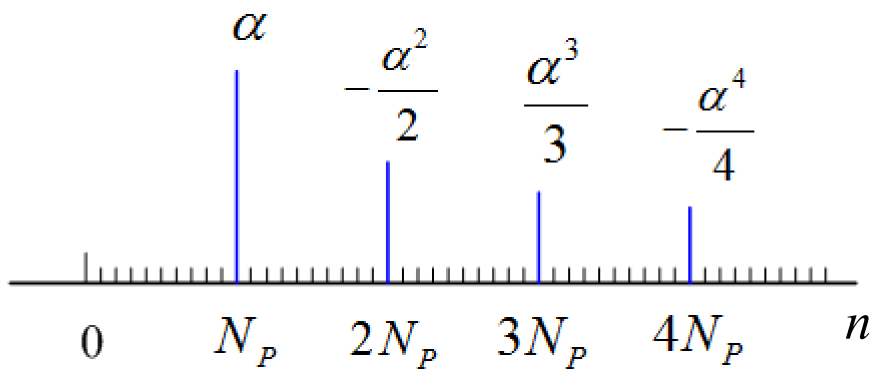
$$y[n] = x[n] + \alpha\, x[n-N_p] = x[n] * p[n]$$

$$P(Z) = 1 + \alpha Z^{-N_p}$$

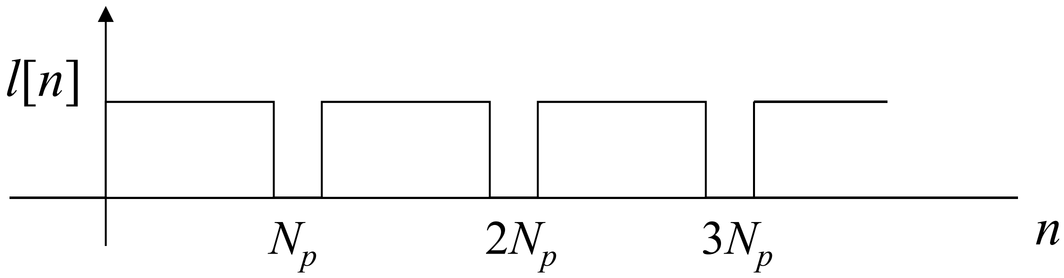
$$\hat{P}(Z) = \log\,(1 + \alpha Z^{-N_p}) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{\alpha^k}{k} Z^{-kN_p}$$

$\downarrow$ Z^{-1}

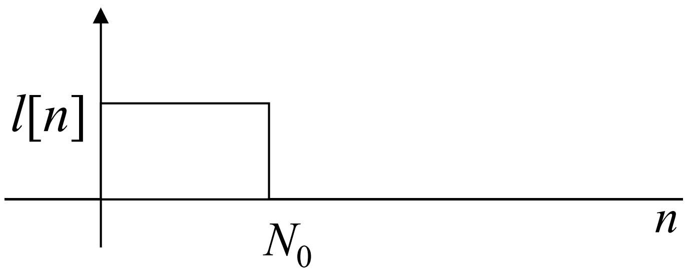
$$\hat{p}[n] = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{\alpha^k}{k} \delta(n - k \cdot N_p)$$



Filtering out the echo by the following "lifter":



or



$Np > N_0$

Q: For the case where N_p is unknown

(2) Representation of acoustic engineering

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

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

(3) Speech analysis

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

Speech wave	Global wave shape	Vocal tract impulse	Pitch

They can be separated by filtering in the complex cepstrum domain

(4) Seismic Signals

(5) Multiple-path analysis for any wave-propagation problem

用 cepstrum 將 multipath的影響去除

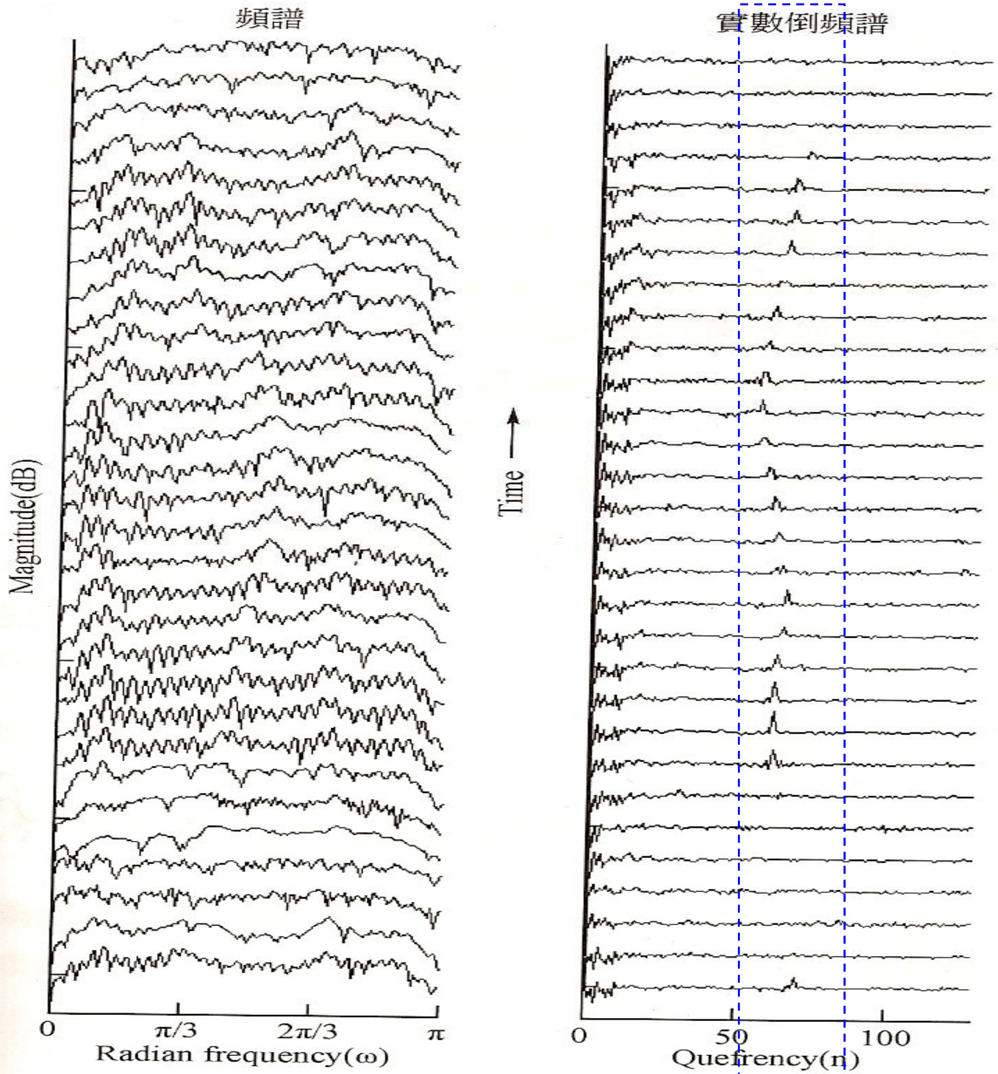


圖 7-5 元音頻譜與倒頻譜

From 王小川，“語音訊號處理”，全華出版，台北，民國94年。

From 王小川，"語音訊號處理"，全華出版，台北，民國94年。

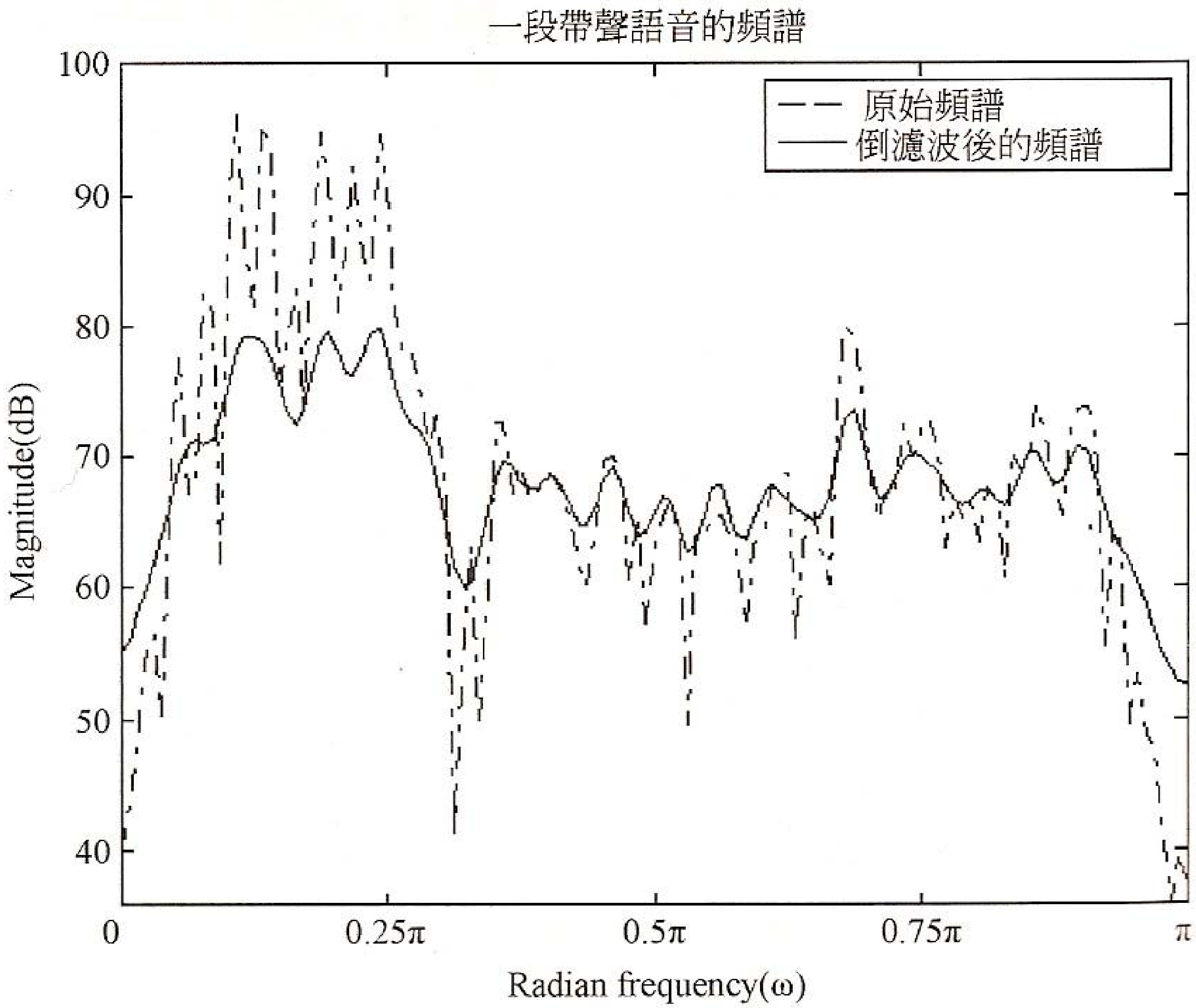


圖 7-6 經過倒濾波器作平滑處理的頻譜

◉ 5-F Problems of Cepstrum

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

(2) Phase

(3) Delay Z^{-k}

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

◉ 5-G Differential Cepstrum

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

inverse Z transform

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

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

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

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

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

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

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

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

$$\hat{y}_d(n) = \hat{x}_d(n) - r\delta(n-1)$$

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

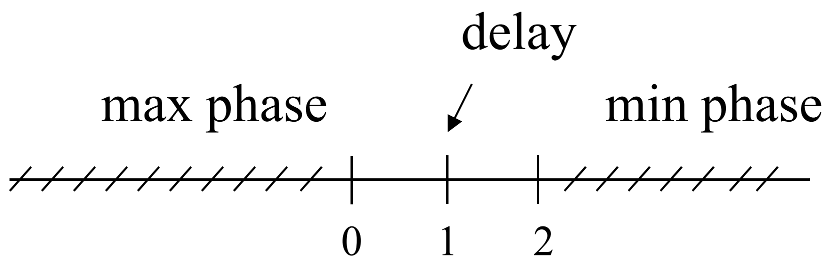
$$\hat{C}(n) = \frac{-\hat{x}_d(n+1)}{n} \qquad n \neq 0 \qquad \textit{diff cepstrum}$$

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

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 $n \ge 2$



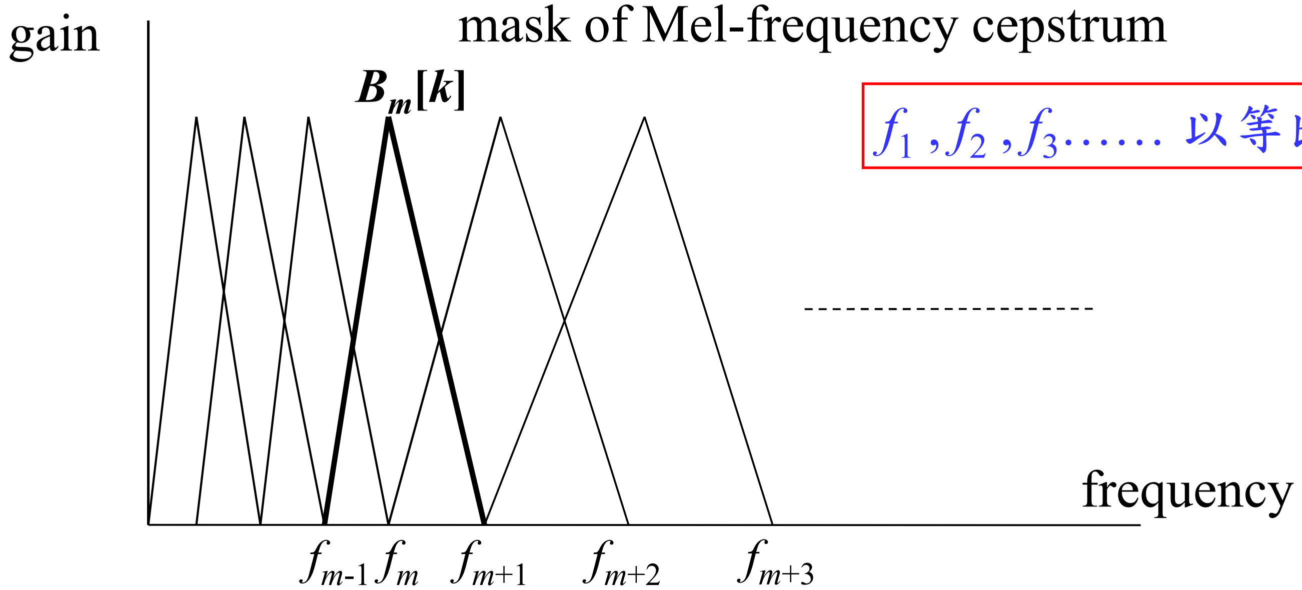
(5) If $x(n)$ is of finite duration, $\hat{x}_d[n]$ has infinite duration

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-H Mel-Frequency Cepstrum (梅爾頻率倒頻譜)

Take log in the frequency mask



$f_1, f_2, f_3 \ldots\ldots$ 以等比級數增加

$f_m = \alpha^m f_0$

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

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

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

$f = k f_s / N$

Process of the Mel-Frequency Cepstrum

summation of the effect inside the m^{th} mask

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

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

Advantages :

Mel-frequency cepstrum 更接近人耳對語音的區別性

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

◎ 5-I 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
  有經過壓縮的聲音檔是*.mp3的型態

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

- 例： [**x**, **fs**] = audioread(C:\WINDOWS\Media\Alarm01.wav');
  可以將 Alarm01.wav 以數字向量 **x** 來呈現。 **fs**: sampling frequency
  這個例子當中 size(**x**) = 122868 2 fs = 22050

- 思考: 所以，取樣間隔多大?

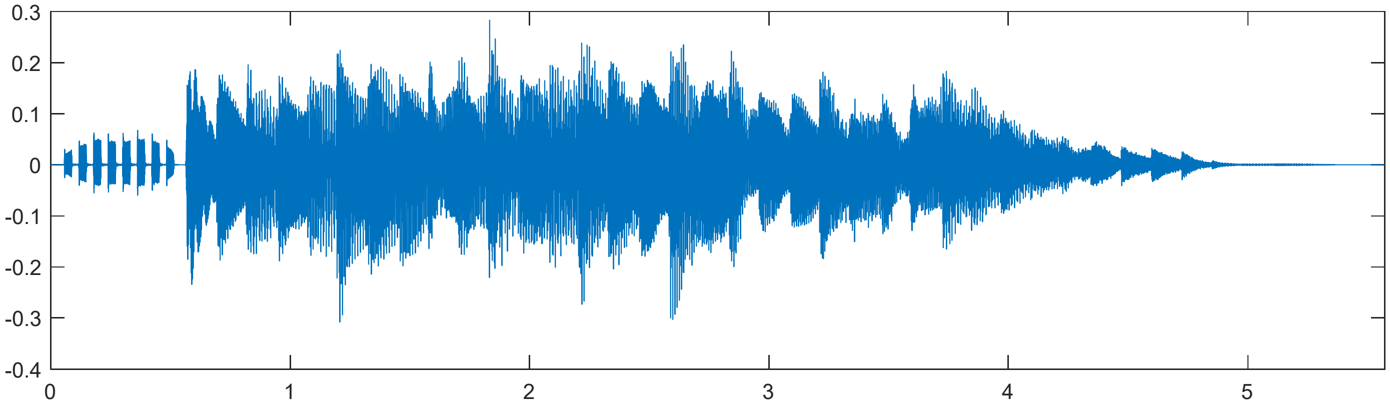
- 這個聲音檔有多少秒？

雙聲道 (**Stereo**，俗稱立體聲)

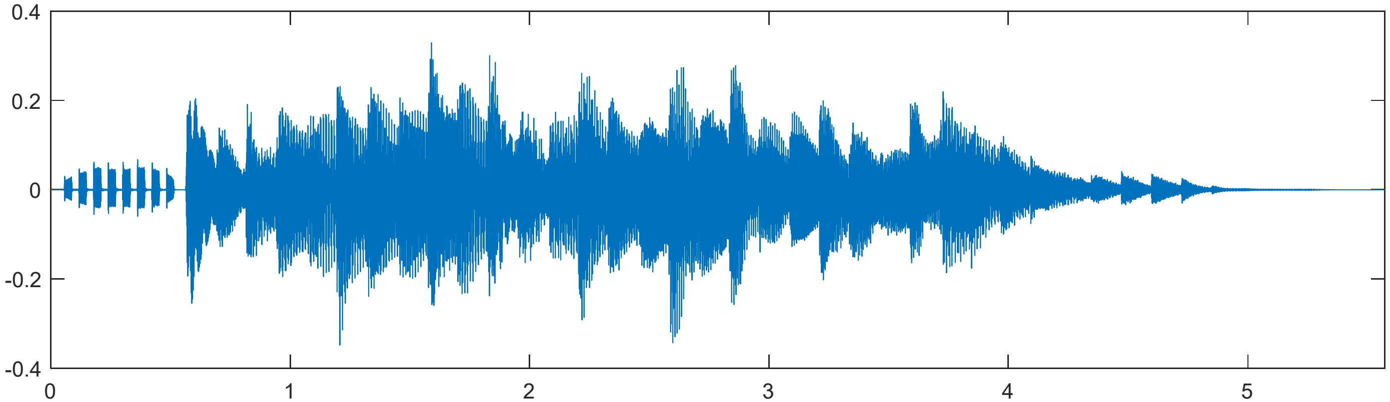
畫出聲音的波型

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

subplot(2,1,1); **plot(time, x(:,1));** xlim([time(1),time(end)])



subplot(2,1,2); **plot(time, x(:,2));** xlim([time(1),time(end)])



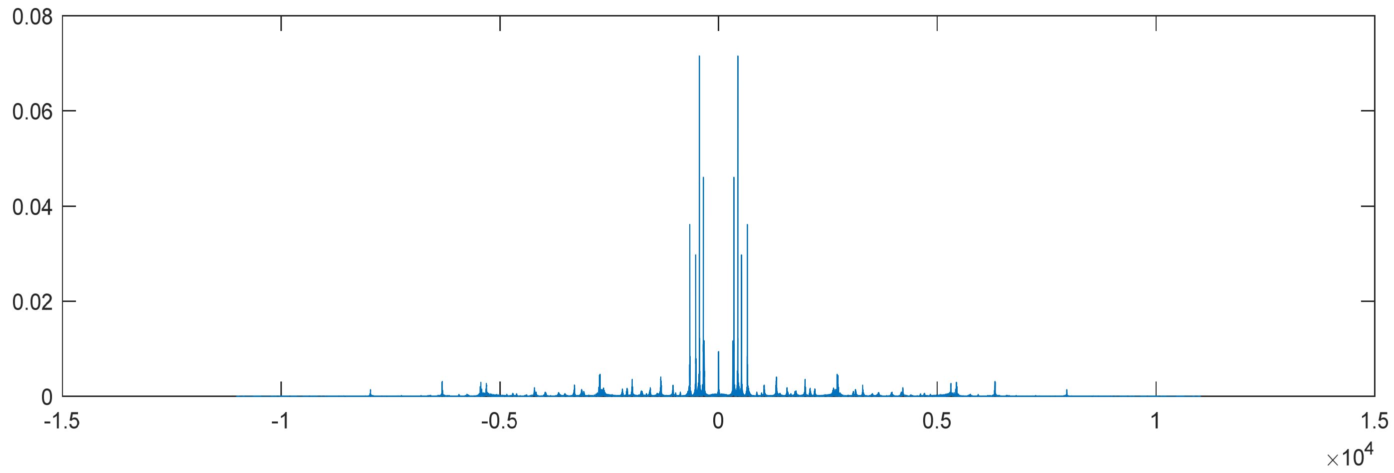
注意： *.wav 檔中所讀取的資料，值都在 −1 和 +1 之間

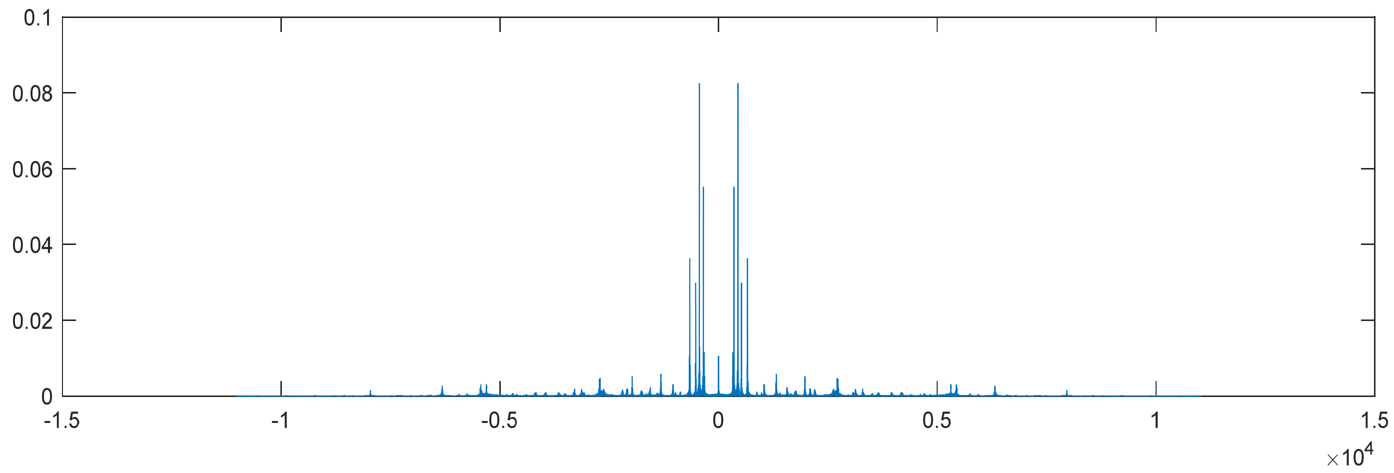
B. 繪出頻譜 (詳細方法請參考附錄二)

X = fft(x(:,1)); % 只做這一步無法得出正確的頻譜

```
X=X.';
N=length(X); N1=round(N/2);
dt=1/fs;
X1=[X(N1+1:N),X(1:N1)]*dt;  % shifting for spectrum
f=[[N1:N-1]-N,0:N1-1]/N*fs;    % valid f
plot(f, abs(X1));
```

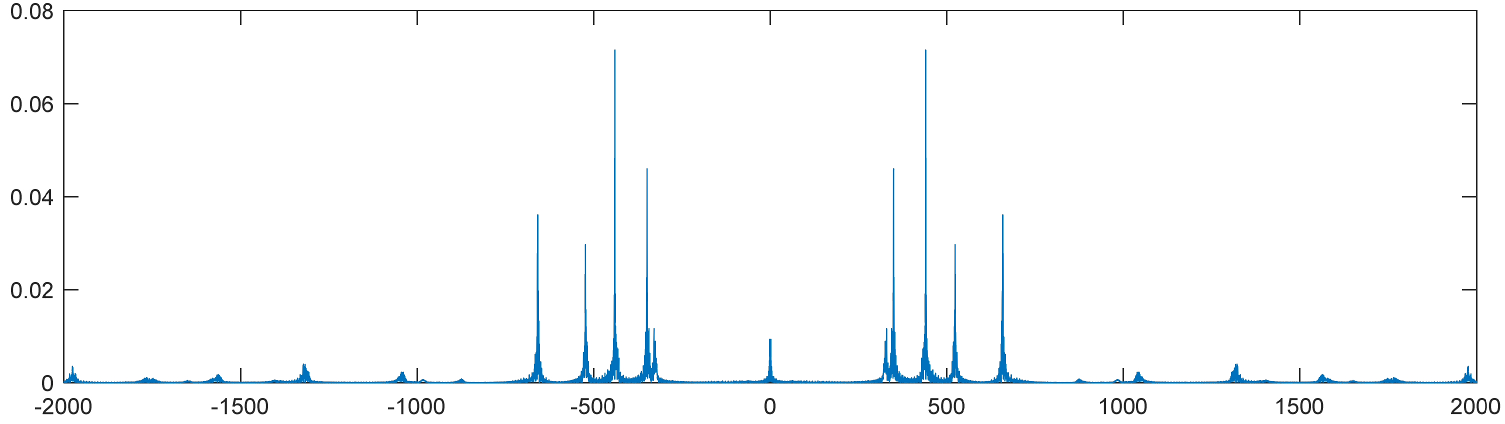
Alarm01.wav 的頻譜

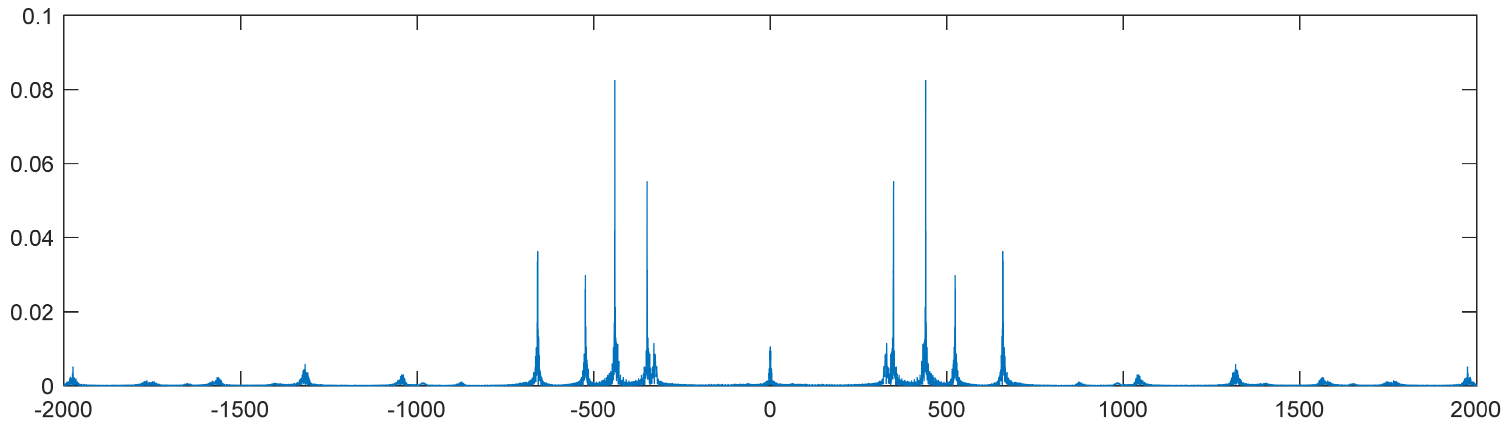




Alarm01.wav 的頻譜

xlim([-2000,2000]) % 只看其中 -2000Hz ~ 2000Hz 的部分





C. 聲音的播放

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

(2) 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

(3) 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： audiowrite

audiowrite(filename, x, fs)

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

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)：

```
sound(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);
audiowrite(‘test.wav’, audioarray, Fs)    % 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：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)

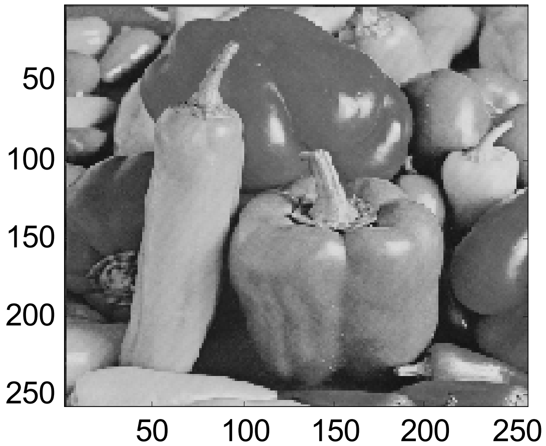
```
im=double(imread('C:\Program Files\MATLAB\pic\Pepper.bmp'));
```

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

size(im2)
```

(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)

```
ans =
  512   512   3
```

```
imshow(im);
```
or
```
image(im/255);
```

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);
```

附錄七 聲音檔和影像檔的處理 (by Python)

可以先安裝幾個模組

```
pip install numpy
pip install scipy
pip install matplotlib  # plot
pip install pipwin
pipwin install simpleaudio  # vocal files
pipwin install pyaudio
```

PS: 謝謝2021年擔任助教的蔡昌廷同學協助製作

A. 讀音訊檔

要先import 相關模組： import scipy.io.wavfile as wavfile

讀取音檔：

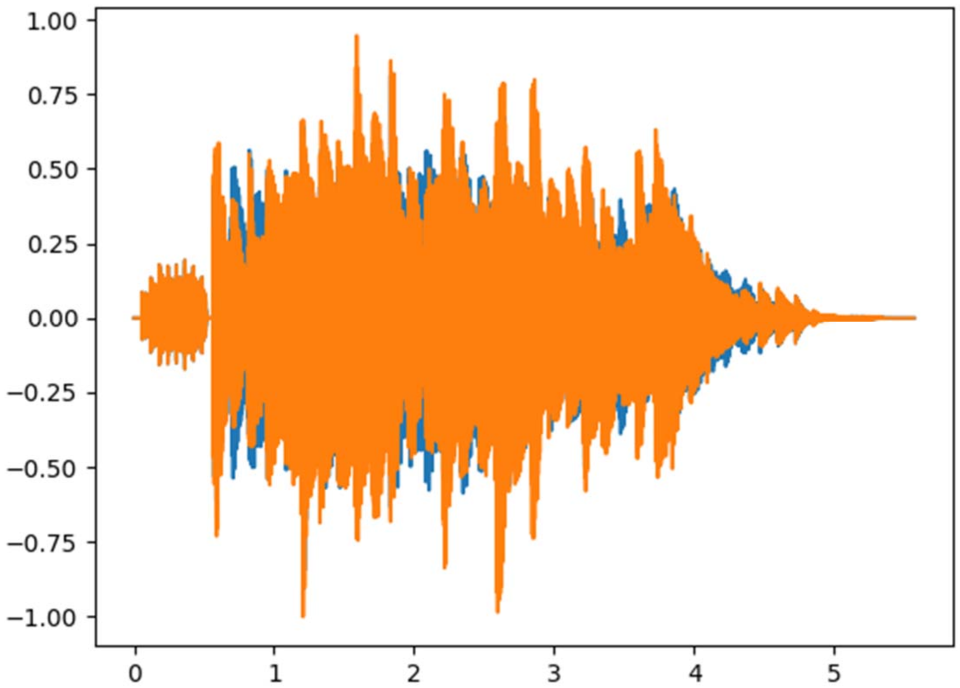
```
fs, wave_data = wavfile.read('C:/WINDOWS/Media/Alarm01.wav’)
  # fs: sampling frequency
  # If the audio file has one channel, then wave_data is a column vector
  # If the audio file has two channels, then wave_data has two column
    vectors
num_frame = len(wave_data)  # 音訊長度:
n_channel = int(wave_data.size/ num_frame)  # channels 數量
```

```
>>> fs
22050
>>> num_frame, n_channel
(1150416, 2)
```

畫出音訊波形圖

要先import 相關模組： import matplotlib.pyplot as plt

- time = np.arange(0, num_frame)*1/fs
- plt.plot(time, wave_data)
- plt.show()



B. 畫出頻譜

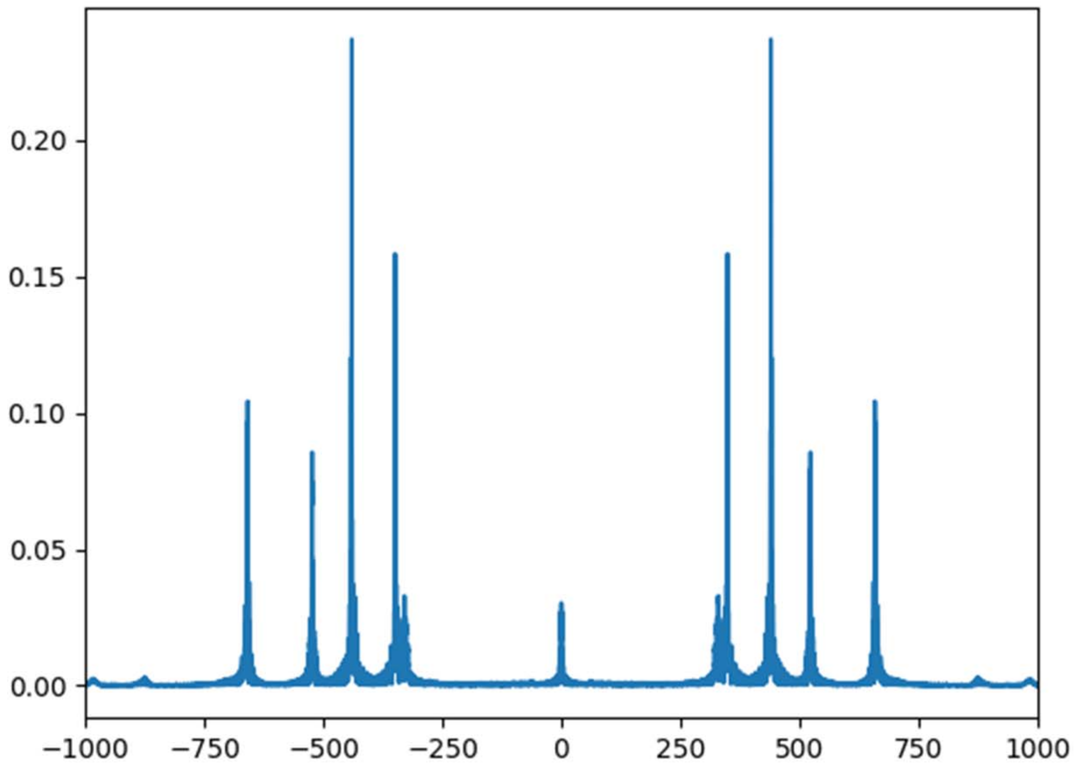
要先import 相關模組： from scipy.fftpack import fft

- fft_data = abs(fft(wave_data[:,1]))/fs   # only choose the 1st channel

  # 注意要乘上 1/fs
- n0=int(np.ceil(num_frame/2))
- fft_data1=np.concatenate([fft_data[n0:num_frame],fft_data[0:n0]])

  # 將頻譜後面一半移到前面
- freq=np.concatenate([range(n0-num_frame,0),range(0,n0)])*fs/num_frame

  # 頻率軸跟著調整
- plt.plot(freq,fft_data1)
- plt.xlim(-1000,1000)   # 限制頻率的顯示範圍
- plt.show() # 如後圖


0.20
0.15
0.10
0.05
0.00
−1000
−750
−500
−250
0
250
500
750
1000

C. 播放聲音

要先import 相關模組： import simpleaudio as sa

- n_bytes =2 # using two bytes to record a data
- wave_data = (2**15-1)* wave_data

  \# change the range to $-2^{15} \sim 2^{15}$
- wave_data = wave_data.astype(np.int16)
- play_obj = sa.play_buffer(wave_data, n_channel, n_bytes, fs)
- play_obj.wait_done()

D. 製作音檔

- wavfile.write(file name, fs, data)

  # fs means the sampling frequency

  # data should be a one-column or two column array

Example:

- wavfile.write('Alarm01_test.wav’, 22050, wave_data)

E. 錄音

要先import 相關模組： import pyaudio

範例程式

```
import pyaudio
pa=pyaudio.PyAudio()
fs = 44100
chunk = 1024
stream = pa.open(format=pyaudio.paInt16, channels=1,
rate=fs, input=True, frames_per_buffer=chunk)

vocal=[]
count=0
```

```
while count<200: #控制錄音時間
   audio = stream.read(chunk)   #一次性錄音取樣位元組大小
   vocal.append(audio)
   count +=1

save_wave_file('testrecord.wav',vocal)
stream.close()
```

參考
https://codertw.com/%E7%A8%8B%E5%BC%8F%E8%AA%9E%E8%A8%80/491427/

F. 影像檔的處理

可以先安裝幾個模組

```
pip install numpy
pip install matplotlib
```

1. 讀取影像檔

```
import cv2
image = cv2.imread('D:/Pic/peppers.bmp’)
```

或

```
import matplotlib.pyplot as plt
image = plt.imread(‘D:/Pic/peppers.bmp’)
```

注意

(1)寫入圖片若為彩色圖片，需要注意 cv2.imread 預設channels 順序為 BGR,

image[:, :, 0] => B, image[:, :, 1] => G, image[:, :, 2] => R

(2) 若使用 plt.imread, 則 3 個 channels 的順序仍為 RGB

image[:, :, 0] => R, image[:, :, 1] => G, image[:, :, 2] => B

(3) 若讀檔讀不出來，有時要將路徑的 \ 改為 /

(4) image.shape 可以看出影像之大小

```
>>> image.shape
(512, 512, 3)
```

(5) 可讀 jpg, bmp, png 檔，但不能讀 gif 檔

2. 顯示影像

Case 1: 圖的值格式為 int

以下的指令要配合使用 (彩色，灰階皆可)

```
cv2.imshow('test', image) # 以 test 為顯示的圖的名稱
cv2.waitKey(0)
cv2.destroyAllWindows()
```

亦可用以下之指令

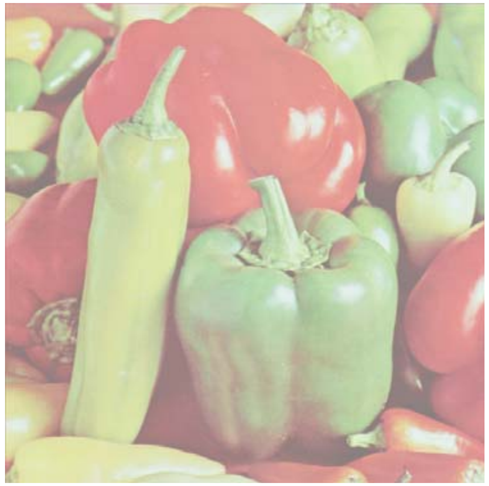
```
import matplotlib.pyplot as plt
plt.imshow(image)
plt.show()
```

若一開始用 cv.imread 讀圖但要用 plt.imshow來顯示彩色圖，要先將 BGR 的順序轉回 RGB，將第二行改為

```
plt.imshow(image[:,:,[2,1,0]])
```

Case 2: 圖的值格式為 double (非整數)

此時，無論用 cv2.imshow或是 plt.imshow，皆要先除以255，再將圖顯示出來

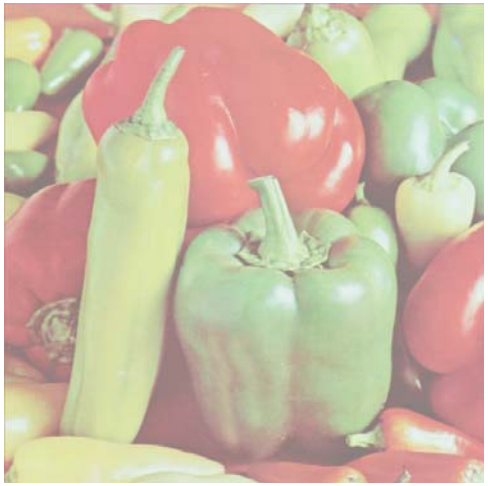


Example 1:

```
image = cv2.imread('D:/Pic/peppers.bmp')
Image1 = image*0.5 + 127.5  # lighten the image
cv2.imshow('test', image) # int 不用除255
cv2.waitKey(0)
cv2.destroyAllWindows()
cv2.imshow('test', image1/255) # 非整數要除255
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Example 2:

```
import matplotlib.pyplot as plt
image = plt.imread('D:/Pic/peppers.bmp’)
image1 = image*0.5 + 127.5  # lighten the image
plt.imshow(image) # int 不用除255
plt.show()
plt.imshow(image1/255) # 非整數要除255
plt.show()
```



3. 寫入圖片檔

cv2.imwrite('D:/Pic/jpg', image)

或

plt.imsave('D:/Pic/jpg', image)

注意

(1) 寫入圖片若為彩色圖片，在使用 cv2.imwrite 時需要注意

image[:, :, 0] => B, image[:, :, 1] => G, image[:, :, 2] => R

若用 plt.imsave 則

image[:, :, 0] => R, image[:, :, 1] => G, image[:, :, 2] => B

(2) 若用 cv2.imwrite(‘D:\Pic\jpg’, image) 可能無法存檔，要將 \ 改為 /

(3) 若是使用 plt.imshow 和 plot.show() 來顯示，可以用右下角的 “save the figure” 來存檔