# Time Frequency Analysis and Wavelet Transforms

# 時頻分析與小波轉換

授課者： 丁 建 均

Office：明達館723室， TEL： 33669652

E-mail: jjding@ntu.edu.tw

課程網頁：http://djj.ee.ntu.edu.tw/TFW.htm

歡迎大家來修課，也歡迎有問題時隨時聯絡！

- **評分方式：**

**平時分數: 15 scores**

基本分11.8分，各位同學皆可拿到(除非缺席狀況嚴重)

另外再根據上課回答問題加分，每回答一次(無論答對否) 加 0.8 分

**Homework: 60 scores**

5 times, 每 3 週一次

請自己寫，和同學內容相同 ，將扣 60% 的分數，就算寫錯但好好寫也會給 40~95% 的分數，遲交分數打 8 折，不交不給分。

不知道如何寫，可用 E-mail 和我聯絡，或於下課時和老師討論

**Term paper 25 scores**

## Term paper 25 scores

方式有四種，可任選其中一種

### (1) 書面報告

(10頁以上(不含封面)，中英文皆可，11或12的字體，題目可選擇和課程有關的任何一個主題。

格式和一般寫期刊論文或碩博士論文相同，包括 abstract, conclusion, 及 references，並且要分 sections，必要時有subsections。 References 的寫法, 可參照一般 IEEE 的論文的寫法 )
鼓勵多做實驗及模擬，有創新更好。

嚴禁剪刀漿糊 (Ctrl-C , Ctrl-V) 的情形，否則扣 60% 的分數

### (2) Tutorial

限十二個名額，和書面報告格式相同，但 18頁以上(若為加強前人的 tutorial，則頁數為 (2/3)$N$ + 13 以上，$N$ 為前人 tutorial 之頁數)，題目由老師指定，以**清楚且有系統**的介紹一個主題的基本概念和應用為要求，為上課內容的進一步探討和補充，**交Word 檔**。

選擇這個項目的同學，學期成績加 3分

## (3) 口頭報告

限四個名額，每個人 40分鐘，題目可選擇和課程有關的任何一個主題。口頭報告將於 12月14日(第 12 週)進行。有意願的同學，請於12月1日之前告知，並附上報告題目。

口頭報告時，鼓勵大家提問（包括口頭報告的同學，也可針對其他同學的報告內容提問）。曾經提問的同學，期末報告皆加2 分。

選擇這個項目的同學，學期成績加2分

## (4) 編輯 Wikipedia

中文或英文網頁皆可，至少 2 個條目，但不可同一個條目翻成中文和英文。總計80行以上。限和課程相關者，自由發揮，越有條理、有系統的越好

選擇編輯 Wikipedia 的同學，請於明年1月4日(本學期最後一次上課)前，向我登記並告知我要編緝的條目(2 個以上)，若有和其他同學選擇相同條目的情形，則較晚向我登記的同學將更換要編緝的條目

書面報告和編輯 Wikipedia，期限是 1月18日

上課時間：17 週

9/14,

9/21,

9/28

10/5, 出 HW1

10/12,

10/19, 交 HW1

10/26, 出 HW2

11/2,

11/9, 交 HW2

11/16, 出 HW3

11/23,

11/30, 交 HW3

12/7, 出 HW4

12/14, Oral

12/2`,

12/28, 交 HW4,

1/4, 出 HW5

1/18, 交 HW5 及 term paper

# 課程大綱：

(1) Introduction

(2) Short-Time Fourier Transform

(3) Gabor Transform

(4) Implementation of Time-Frequency Analysis

(5) Wigner Distribution Function

(6) Cohen’s Class Time-Frequency Distribution

(7) S Transforms, Gabor-Wigner Transforms, Matching Pursuit, and Other Time Frequency Analysis Methods

(8) Movement in the Time-Frequency Plane and Fractional Fourier Transforms

(9) Filter Design by Time-Frequency Analysis

(10) Modulation, Multiplexing, Sampling, and Other Applications

(續)

## 課程大綱：

(11) Hilbert Huang Transform

(12) From Haar Transforms to Wavelet Transforms

(13) Continuous Wavelet Transforms

(14) Continuous Wavelet Transform with Discrete Coefficients

(15) Discrete Wavelet Transform

(16) Applications of the Wavelet Transform

- **上課資料：**

**(1) 講義 (將放在網頁上，請大家每次上課前先印好)**

(2) S. Qian and D. Chen, *Joint Time-Frequency Analysis: Methods and Applications*, Prentice-Hall, 1996.

(3) L. Cohen, *Time-Frequency Analysis*, Prentice-Hall, New York, 1995.

(4) K. Grochenig, *Foundations of Time-Frequency Analysis*, Birkhauser, Boston, 2001.

(5) L. Debnath, *Wavelet Transforms and Time-Frequency Signal Analysis*, Birkhäuser, Boston, 2001.

(6) S. Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*, Academic Press, 3rd ed., 2009.

(7) Others

## Matlab Program

Download: 請洽台大電信所

http://comm.ntu.edu.tw/matlab/request.php

參考書目

洪維恩，Matlab 7 程式設計，旗標，台北市，2010. .  (合適的入門書)

張智星，Matlab 程式設計入門篇，第三版，碁峰，2011.

蒙以正，數位信號處理：應用 Matlab，旗標，台北市，2007.

繆紹綱譯，數位影像處理：運用-Matlab，東華，2005.

預計看書學習所花時間： 3~5 天

**Tutorial 可供選擇的題目(可以略做修改)**

有加 * 號的是學長寫過的，但鼓勵同學再加強

(1) Time-Frequency Reassignment

(2) Sparse Time-Frequency Representation

(3) Fast Algorithm for Time-Frequency Analysis

(4) Time-Frequency Analysis for Machine Fault Detection

(5) Orthogonal Matching Pursuit

(6) Compressive Sensing for Signal Reconstruction

(7) Compressive Sensing for Radar Imaging

(8) Compressive Sensing for Communication

(9) Compressive Sensing for Denoising

(10) Hilbert-Huang Transform for EMG Signal Processing

(11) Wavelet Transforms for Image Feature Extraction

(12) Wavelet Transforms for Watermarking

# I. Introduction

**Fourier transform (FT)**

$X(f)=\int_{-\infty}^{\infty}x(t)e^{-j2\pi f t}dt$ Time-Domain $\rightarrow$ Frequency Domain

↑ $t$ varies from $-\infty\sim\infty$

Laplace Transform $X(s)=\int_{-\infty}^{\infty}x(t)e^{-st}dt$

Cosine Transform, Sine Transform, Z Transform.

Some things make these operations not practical:

(1) Only the case where $t_0 \le t \le t_1$ is interested.

(2) Not all the signals are suitable for analyzing in the frequency domain.

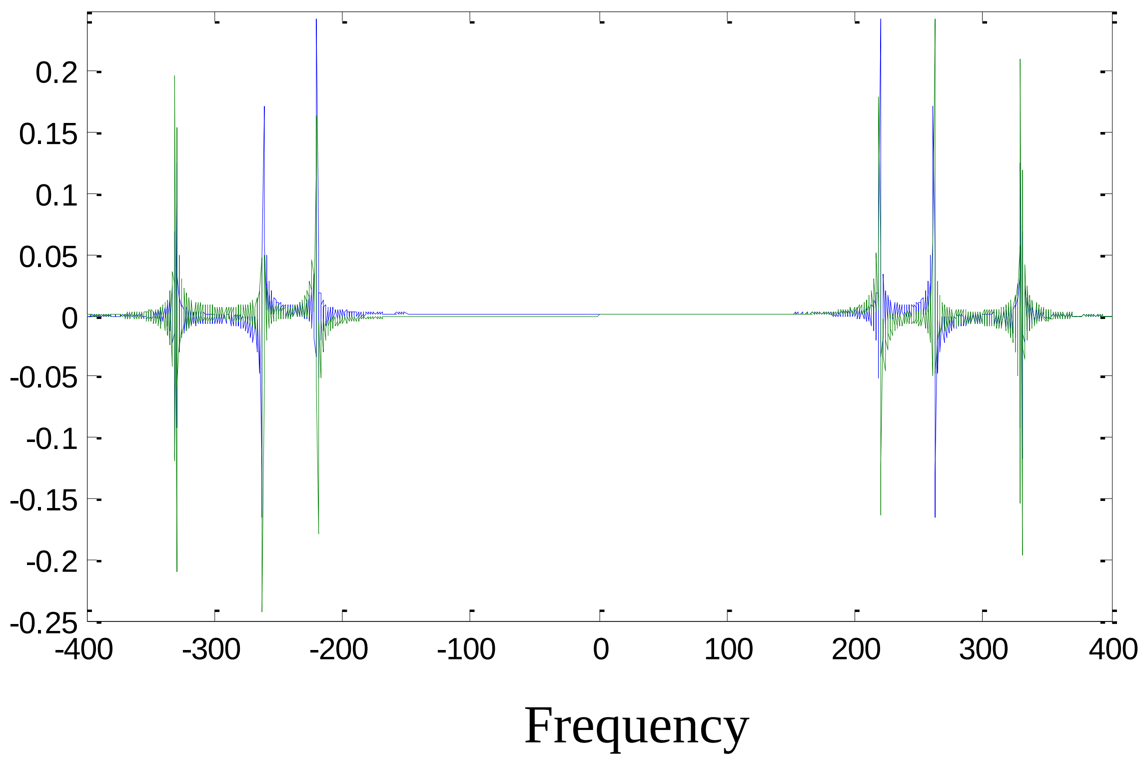
It is hard to observe the variation of spectrum with time by these operations

Example 1: $x(t) = \cos(440\pi t)$ when $t < 0.5$,

$x(t) = \cos(660\pi t)$ when $0.5 \le t < 1$,

$x(t) = \cos(524\pi t)$ when $t \ge 1$

The Fourier transform of $x(t)$



**(A) Finite-Supporting Fourier Transform**

$$X(f) = \int_{t_0 - B}^{t_0 + B} x(t) e^{-j2\pi f t} dt$$

**(B) Short-Time Fourier Transform (STFT)**

$$X(t, f) = \int_{-\infty}^{\infty} w(t - \tau) x(\tau) e^{-j2\pi f \tau} d\tau$$

$w(t)$: window function 或 mask function

STFT 也稱作 windowed Fourier transform 或

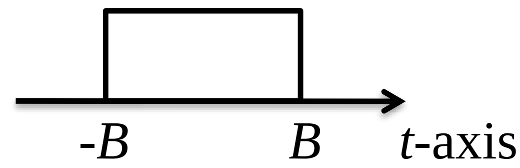
time-dependent Fourier transform

[Ref] L. Cohen, *Time-Frequency Analysis*, Prentice-Hall, New York, 1995.

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

最簡單的例子： $w(t) = 1$ for $|t| \le B$,

$w(t) = 0$ otherwise

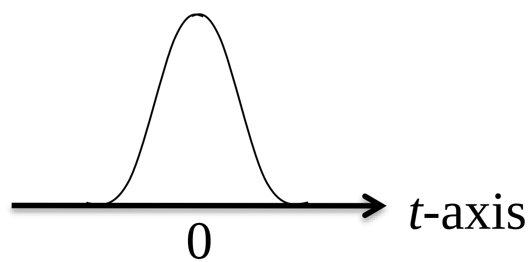



此時 Short-time Fourier transform 可以改寫

$$X(t, f) = \int_{t-B}^{t+B} x(\tau) e^{-j2\pi f \tau} d\tau$$

其他的例子：

$$w(t) = \exp\left(-\sigma t^2\right)$$




一般我們把 exp(- $\sigma t^2$) 稱作為 Gaussian function 或 Gabor function

此時的 Short-Time Fourier Transform 亦稱作 Gabor Transform

**(C) Gabor Transform**

$$G_x(t, f) = \int_{-\infty}^{\infty} e^{-\pi\sigma(\tau-t)^2} e^{-j2\pi f\tau} x(\tau)\,d\tau$$

$$G_x(t,\omega) = \sqrt{\frac{1}{2\pi}} \int_{-\infty}^{\infty} e^{-\frac{\sigma(\tau-t)^2}{2}} e^{-j\omega\tau} x(\tau)\,d\tau$$

- S. Qian and D. Chen, *Joint Time-Frequency Analysis: Methods and Applications*,

  Prentice Hall, N.J., 1996.

- R. L. Allen and D. W. Mills, *Signal Analysis: Time, Frequency, Scale, and*

  *Structure*, Wiley- Interscience.

Common Features for short-time Fourier transforms and Gabor transforms

(1) The instantaneous frequency can be observed

(2) Without Cross Term

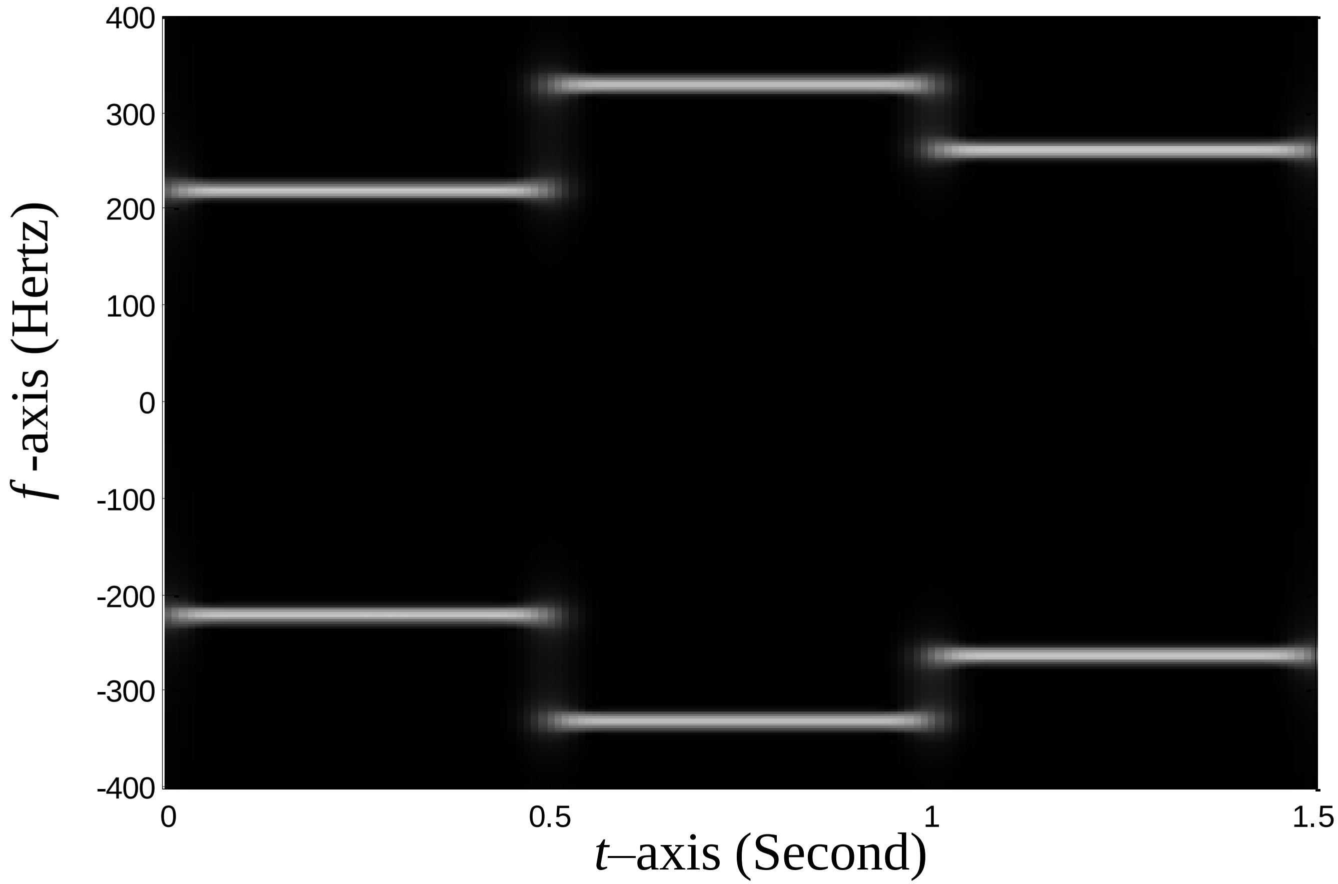
(3) Poor clarity

Example: $x(t) = \cos(440\pi t)$ when $t < 0.5$,

$x(t) = \cos(660\pi t)$ when $0.5 \le t < 1$,

$x(t) = \cos(524\pi t)$ when $t \ge 1$

The Gabor transform of $x(t)$ ($\sigma = 200$)




用 Gray level 來表示 $X(t, f)$ 的 amplitude

## Instantaneous Frequency 瞬時頻率

If $x(t) = \sum_{k=1}^{N} a_k \cdot \exp(j \cdot \phi_k(t))$ around $t_0$

then the instantaneous frequency of $x(t)$ at $t_0$ are

$$\frac{\phi_1'(t_0)}{2\pi}, \frac{\phi_2'(t_0)}{2\pi}, \frac{\phi_3'(t_0)}{2\pi}, \cdots, \frac{\phi_N'(t_0)}{2\pi}$$ (以頻率 frequency 表示)

$$\phi_1'(t_0), \phi_2'(t_0), \phi_3'(t_0), \cdots, \phi_N'(t_0)$$ (以角頻率 angular frequency 表示)：

If the order of $\phi_k(t)$ > 1, then instantaneous frequency varies with time

## 自然界中，頻率會隨著時間而改變的例子

Frequency Modulation

Music

Speech

Others (Animal voice, Doppler effect, seismic waves, radar system, optics, rectangular function)

In fact, in addition to **sinusoid-like functions**, the instantaneous frequencies of other functions will inevitably vary with time.

- Sinusoid Function

- Chirp function

$$\exp\left[\, j(\alpha_2 t^2 + \alpha_1 t + \alpha_0) \right] \qquad \text{Instantaneous frequency} = \frac{\alpha_2}{\pi} t + \frac{\alpha_1}{2\pi}$$

acoustics, wireless communication, radar system, optics

例： ㄚ ($F_1$ = 900Hz, $F_2$ = 1200Hz) , ㄧ ($F_1$ = 300Hz, $F_2$ = 2300Hz)

$F_1$ 由嘴唇的大小決定, $F_2$ - $F_1$ 由如面的高低決定

- Higher order exponential function

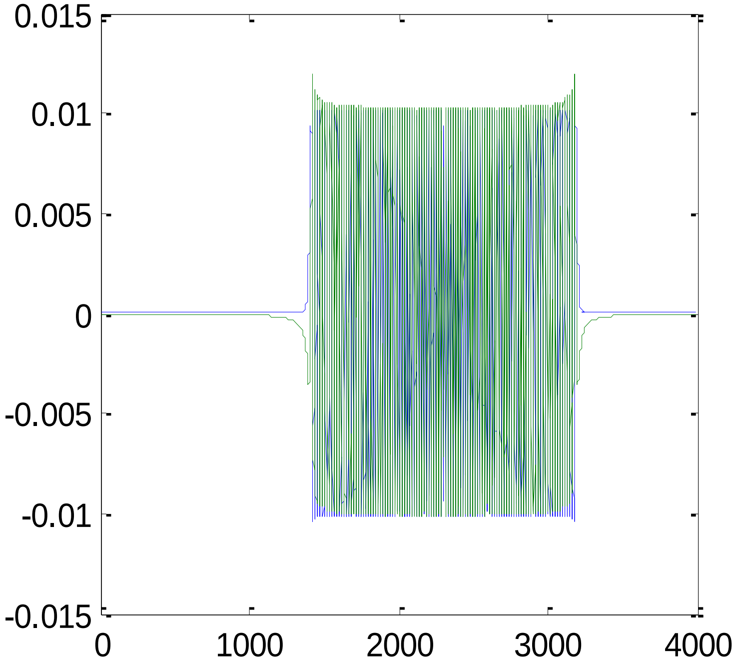
Example 2

(1) $x(t) = 0.5\cos\left(6400\pi t - 600\pi t^2\right)$ $t \in [0, 3]$

(2) $x(t) = 0.5\cos\left(600\pi t^3 - 2700\pi t^2 + 5050\pi t\right)$ $t \in [0, 3]$

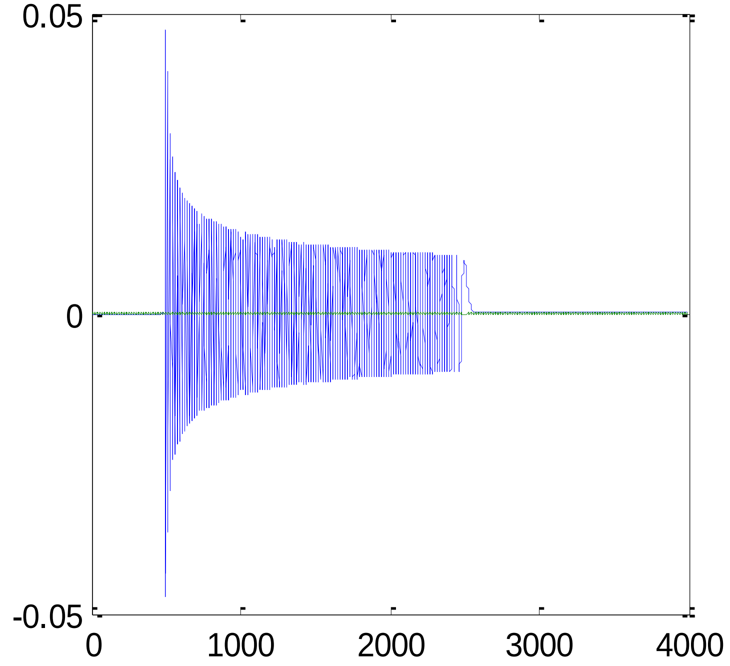
# Fourier transform

$$x(t) = 0.5\cos\left(6400\pi t - 600\pi t^2\right)$$




$f$ (Hz)

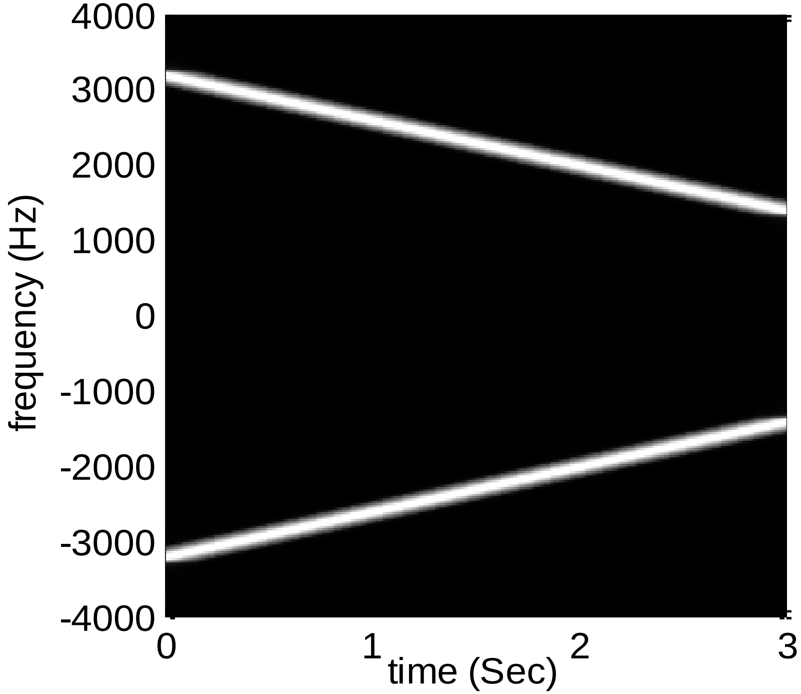
$$x(t) = 0.5\cos\left(600\pi t^3 - 2700\pi t^2 + 5050\pi t\right)$$




$f$ (Hz)

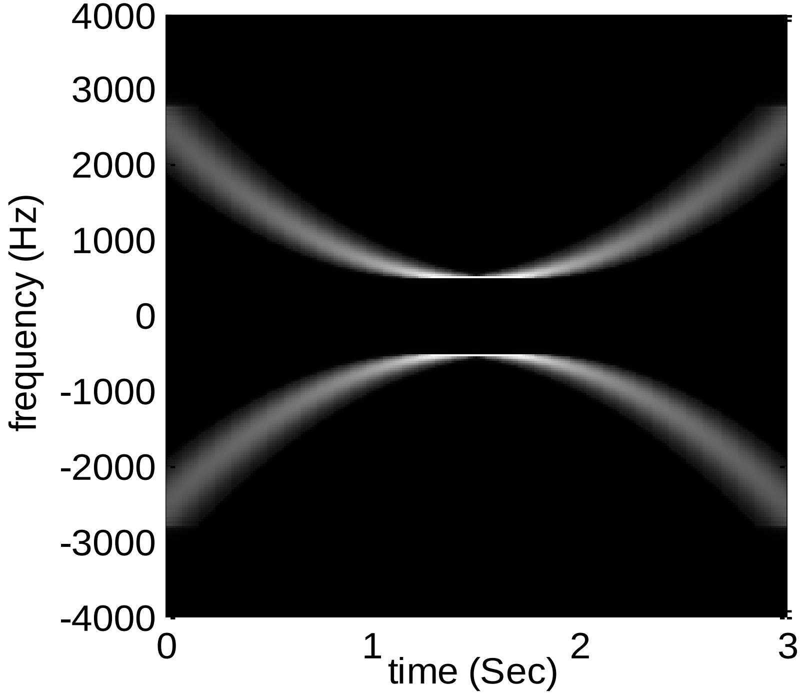
(1)

$$x(t) = 0.5\cos\left(6400\pi t - 600\pi t^2\right)$$




(2)

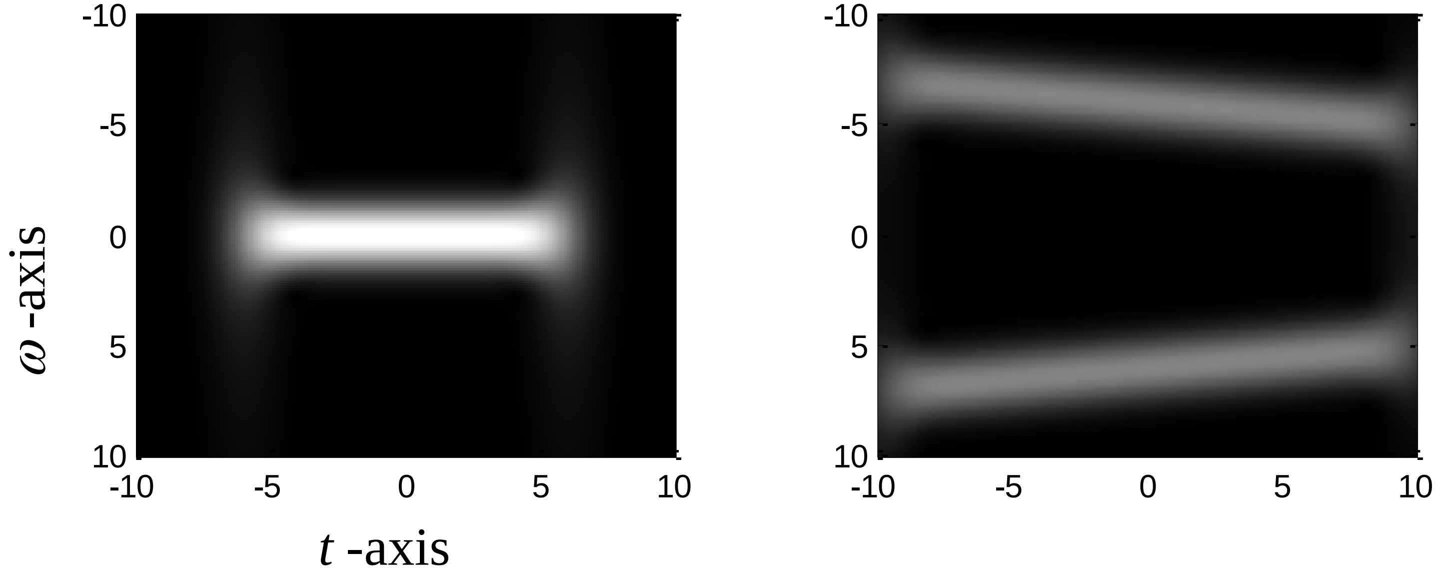
$$x(t) = 0.5\cos\left(600\pi t^3 - 2700\pi t^2 + 5050\pi t\right)$$



# Example 3

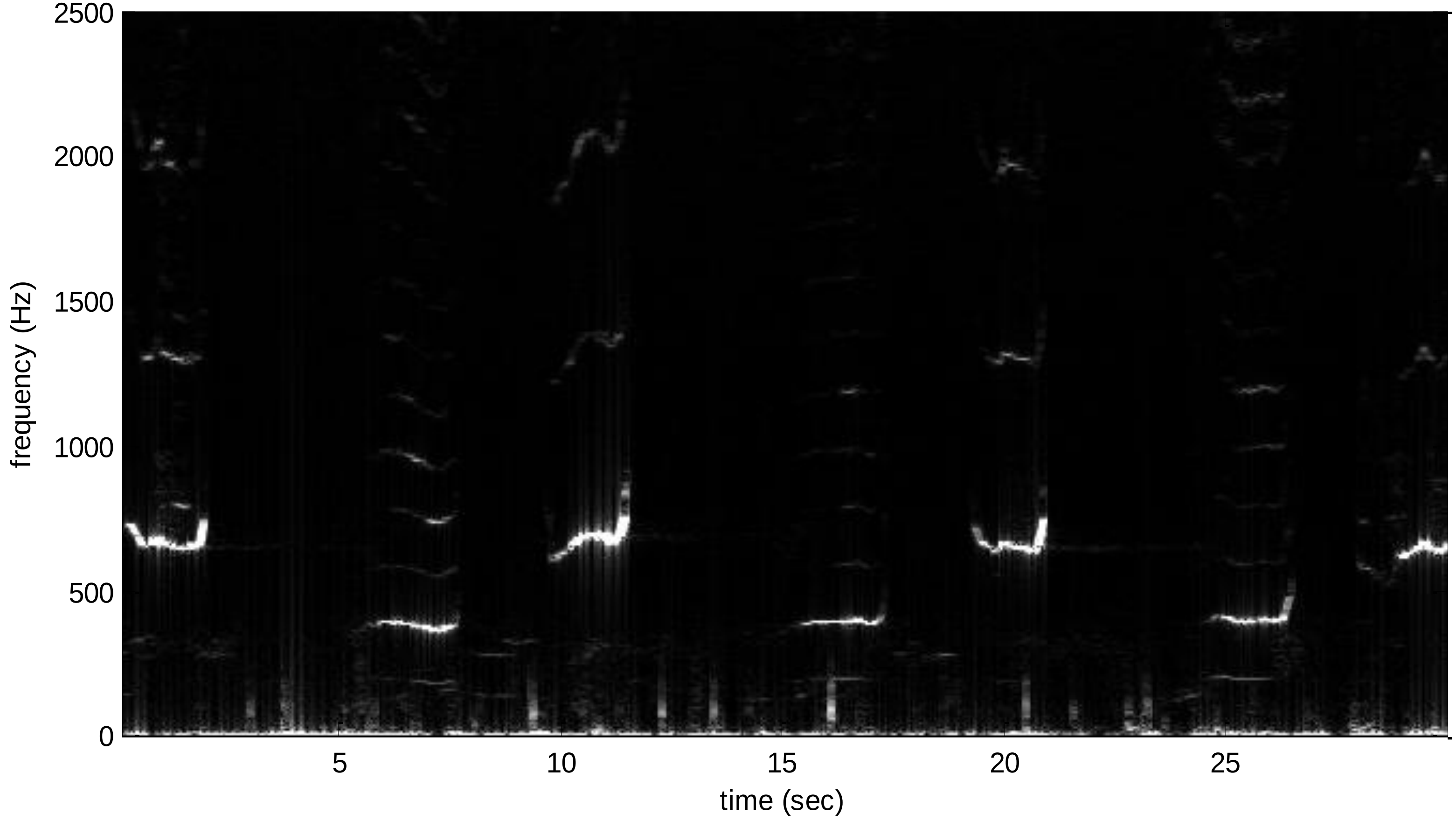
left: $x_1(t) = 1$ for $|t| \leq 6$, $x_1(t) = 0$ otherwise, right: $x_2(t) = \cos(6t - 0.05t^2)$

Gabor transform



# Example 4

Data source: http://oalib.hlsresearch.com/Whales/index.html



# Why Time-Frequency Analysis is Important?

- Many digital signal processing applications are related to the spectrum or the bandwidth of a signal.

- If the spectrum and the bandwidth can be determined adaptive, the performance can be improved.

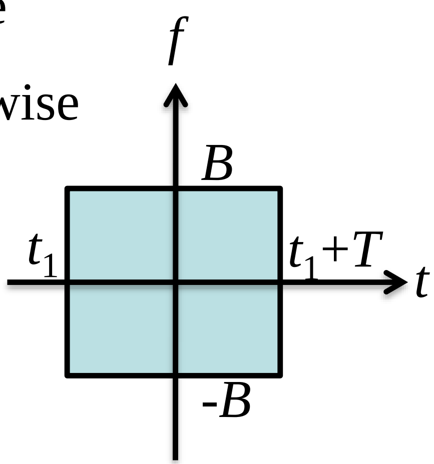
- modulation,
- multiplexing,
- filter design,
- data compression,
- signal analysis,
- signal identification,
- acoustics,
- system modeling,
- radar system analysis
- sampling

## Example: Generalization for sampling theory

假設有一個信號，

① The supporting of $x(t)$ is $t_1 \le t \le t_1 + T$, $x(t) \approx 0$ otherwise

② The supporting of $X(f) \neq 0$ is $-B \le f \le B$, $X(f) \approx 0$ otherwise




根據取樣定理， $\Delta_t \le 1/F$ , $F = 2B$, $B$:頻寬

所以，取樣點數 $N$ 的範圍是

$$N = T/\Delta_t \ge TF$$

**重要定理：**一個信號所需要的取樣點數的下限，等於它時頻分佈的面積

Q1：Scaling 對於一個信號的取樣點數有沒有影響？
Hint:

$$g(\sigma t) \xrightarrow{FT} \frac{1}{|\sigma|} G\left(\frac{f}{\sigma}\right)$$

Q2: How to use time-frequency analysis to reduce the number of sampling points?

Time-frequency analysis is an efficient tool for adaptive signal processing.

# 時頻分析的大家族

- (1) Short-time Fourier transform (STFT) (rec-STFT, Gabor, …)
  - square → spectrogram
  - improve → Asymmetric STFT
  - improve → S transform
  - combine → **Gabor-Wigner Transform**
- (2) Wigner distribution function (WDF)
  - combine → **Gabor-Wigner Transform**
  - improve → windowed WDF
  - improve → Cohen's Class Distribution (Choi-Williams, Cone-Shape, Page, Levin, Kirkwood, Born-Jordan, …)
  - improve → Pseudo L-Wigner Distribution
- (3) Wavelet transform
  - → Haar and Daubechies
  - → Coiflet, Morlet
  - → Directional Wavelet Transform
- (4) Time-Variant Basis Expansion
  - → Matching Pursuit
  - → Prolate Spheroidal Wave Function
- (5) Hilbert-Huang Transform (唯一跳脫 Fourier transform 的架構)

# • Continuous Wavelet Transform

forward wavelet transform:

$$X(a,b) = \frac{1}{\sqrt{b}} \int_{-\infty}^{\infty} x(t)\psi\left(\frac{t-a}{b}\right)dt$$

$\psi(t)$: mother wavelet, $a$: location, $b$: scaling,

inverse wavelet transform:

$$x(t) = \sum_{a}\sum_{b} X(a,b)\varphi_{a,b}(t)$$

$\varphi_{a,b}(t)$ is dual orthogonal to $\psi(t)$.

| | output |
|---|---|
| Fourier transform | $X(f)$, $f$: frequency |
| time-frequency analysis | $X(t, f)$, $t$: time, $f$: frequency |
| wavelet transform | $X(a, b)$, $a$: time, $b$: scaling |

限制：

(1) $$\frac{1}{\sqrt{b}}\int_{-\infty}^{\infty}\varphi_{a_1,b_1}(t)\psi\left(\frac{t-a}{b}\right)dt=1$$ when $a_1 = a$ and $b_1 = b$,

$$\frac{1}{\sqrt{b}}\int_{-\infty}^{\infty}\varphi_{a_1,b_1}(t)\psi\left(\frac{t-a}{b}\right)dt=0$$ otherwise

(2) $\psi(t)$ has a finite time interval

Two parameters, $a$: 調整位置, $b$: 調整寬度

應用： adaptive signal analysis

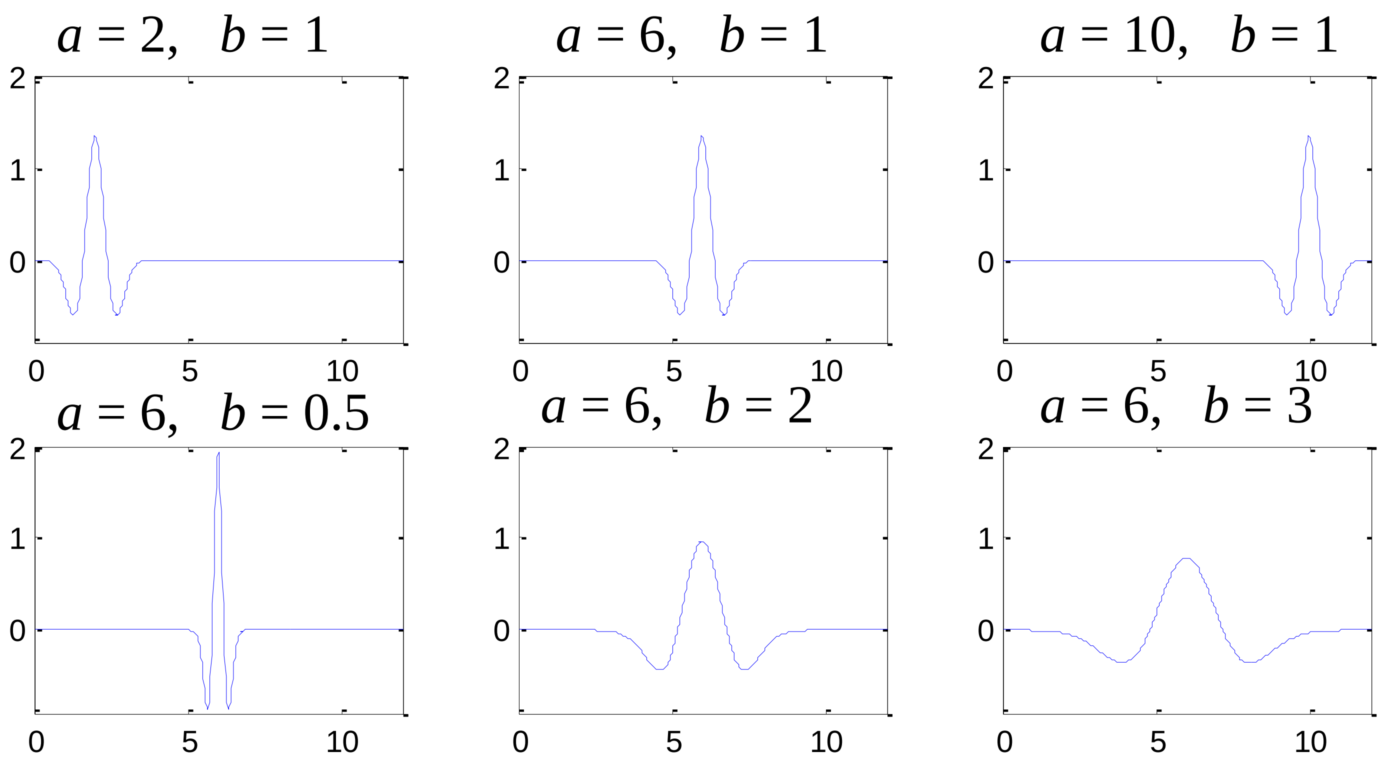
思考：需要較高解析度的地方，$b$ 的值應該如何？

Wavelet 的種類甚多

Mexican hat wavelet, Haar Wavelet, Daubechies wavelet, triangular wavelet, …………

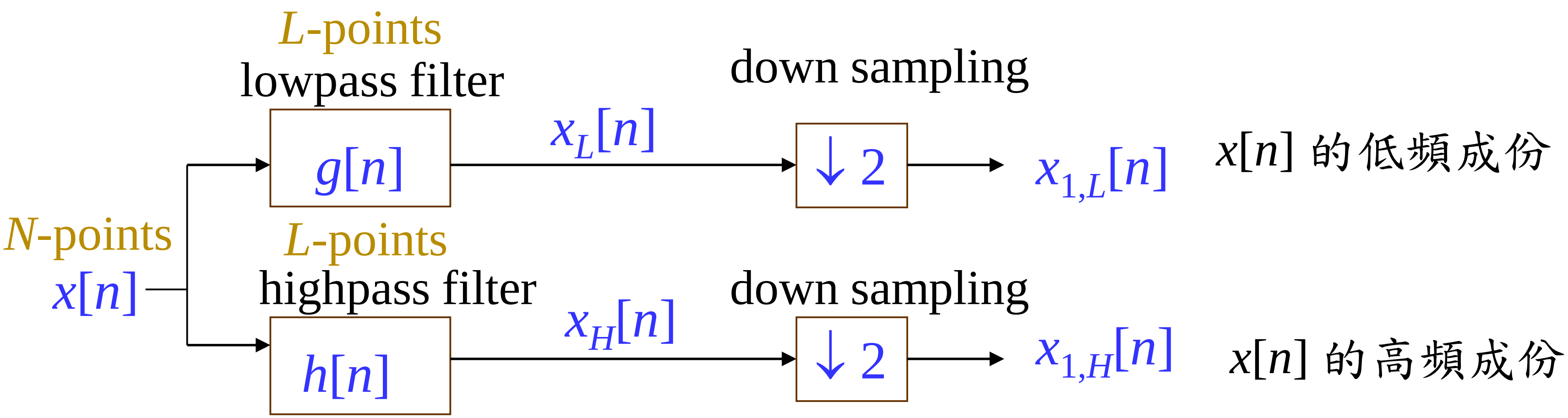
例子： Mexican hat wavelet 隨 $a$ and $b$ 變化之情形

$$\psi(t)=\frac{2^{5/4}}{\sqrt{3}}\left(1-2\pi t^2\right)e^{-\pi t^2} \qquad \frac{1}{\sqrt{b}}\psi\left(\frac{t-a}{b}\right)$$



# • Discrete Wavelet Transform (DWT)

The discrete wavelet transform is very different from the continuous wavelet transform. It is simpler and more useful than the continuous one.




$$x_L[n] = \sum_k x[n-k]g[k] \qquad x_{1,L}[n] = \sum_k x[2n-k]g[k]$$

$$x_H[n] = \sum_k x[n-k]h[k] \qquad x_{1,H}[n] = \sum_k x[2n-k]h[k]$$

$$x_{1,L}[n] = \sum_k x[2n-k]g[k] \qquad x_{1,H}[n] = \sum_k x[2n-k]h[k]$$

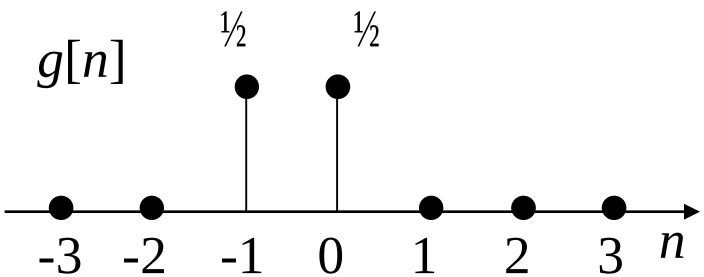
例子：2-point Haar wavelet

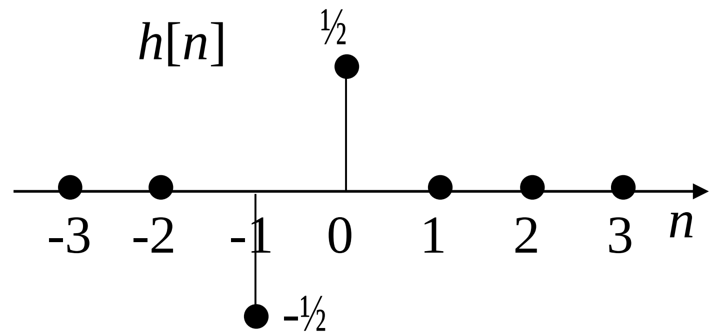
$g[n] = 1/2$ for $n = -1, 0$

$g[n] = 0$ otherwise

$h[0] = 1/2, \quad h[-1] = -1/2,$

$h[n] = 0$ otherwise




then

$$x_{1,L}[n] = \frac{x[2n] + x[2n+1]}{2}$$

（兩點平均）

$$x_{1,H}[n] = \frac{x[2n] - x[2n+1]}{2}$$

（兩點之差）

Discrete wavelet transform 有很多種

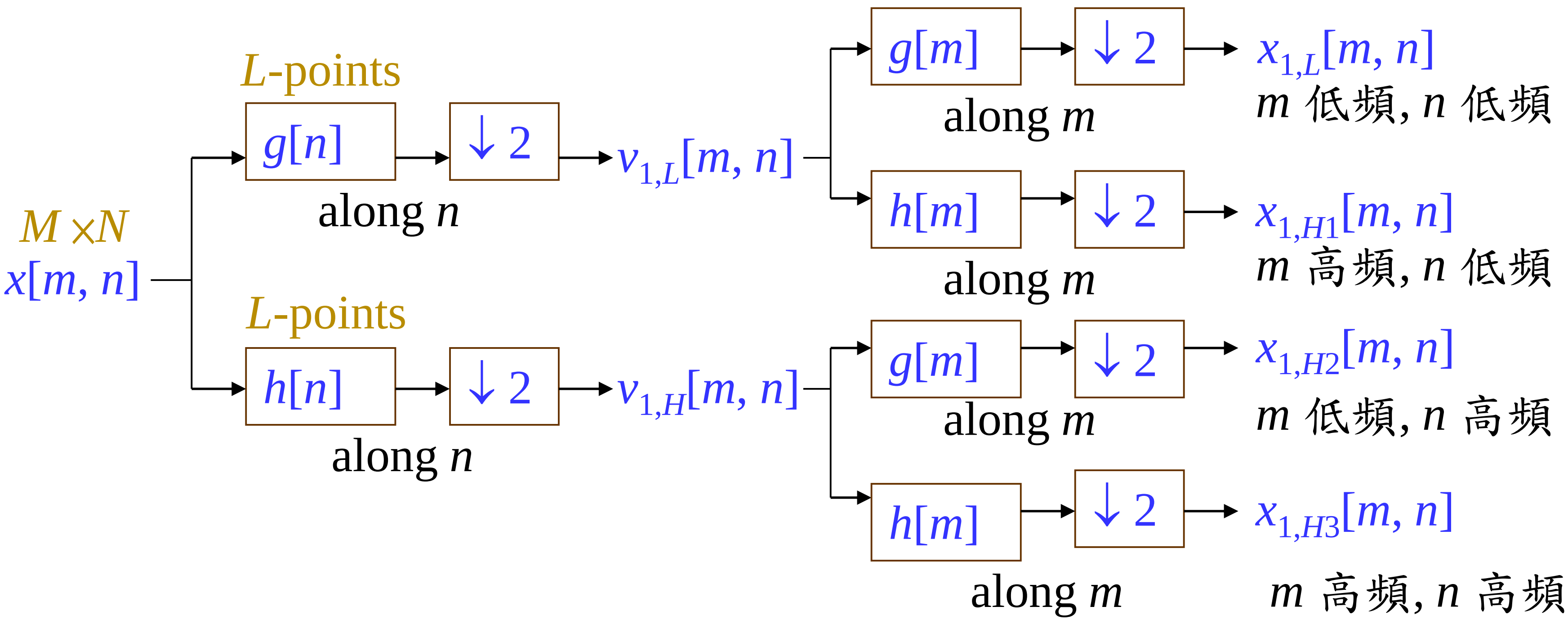
(discrete Haar wavelet, discrete Daubechies wavelet, B-spline DWT, symlet, coilet, ……..)

一般的 wavelet, $g[n]$ 和 $h[n]$ 點數會多於 2 點

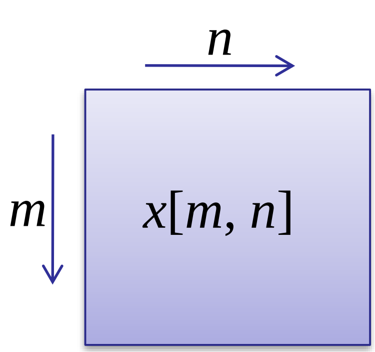
但是 $g[n]$ 通常都是 lowpass filter 的型態

$h[n]$ 通常都是 highpass filter 的型態

# 2-D 的情形



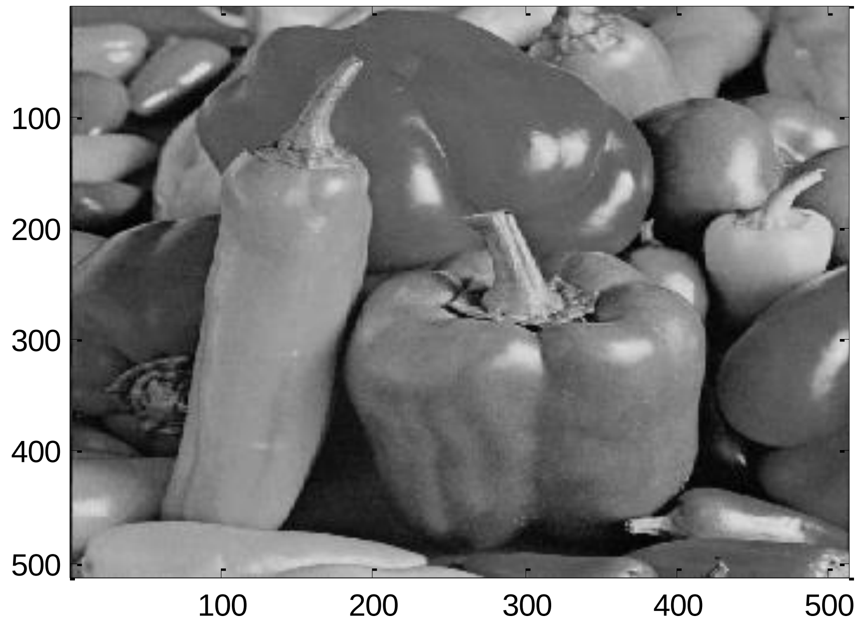
L-points
g[n]
↓2
along n
v1,L[m, n]
M × N
x[m, n]
L-points
h[n]
↓2
along n
v1,H[m, n]
g[m]
↓2
along m
x1,L[m, n]
m 低頻, n 低頻
h[m]
↓2
along m
x1,H1[m, n]
m 高頻, n 低頻
g[m]
↓2
along m
x1,H2[m, n]
m 低頻, n 高頻
h[m]
↓2
along m
x1,H3[m, n]
m 高頻, n 高頻

n
m
x[m, n]

原影像

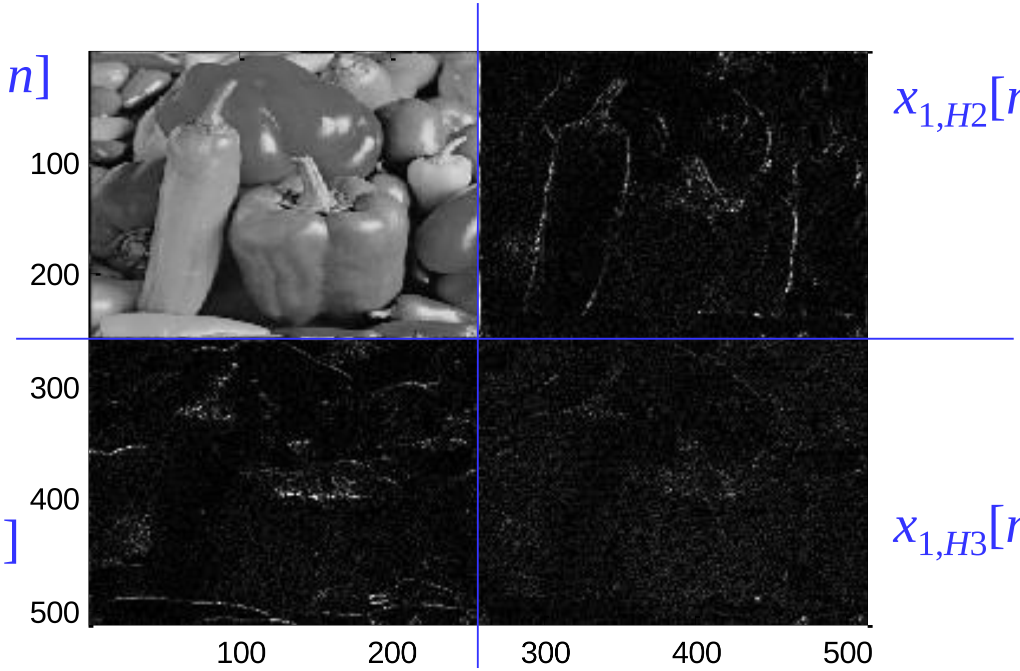
Pepper.bmp




2-D DWT

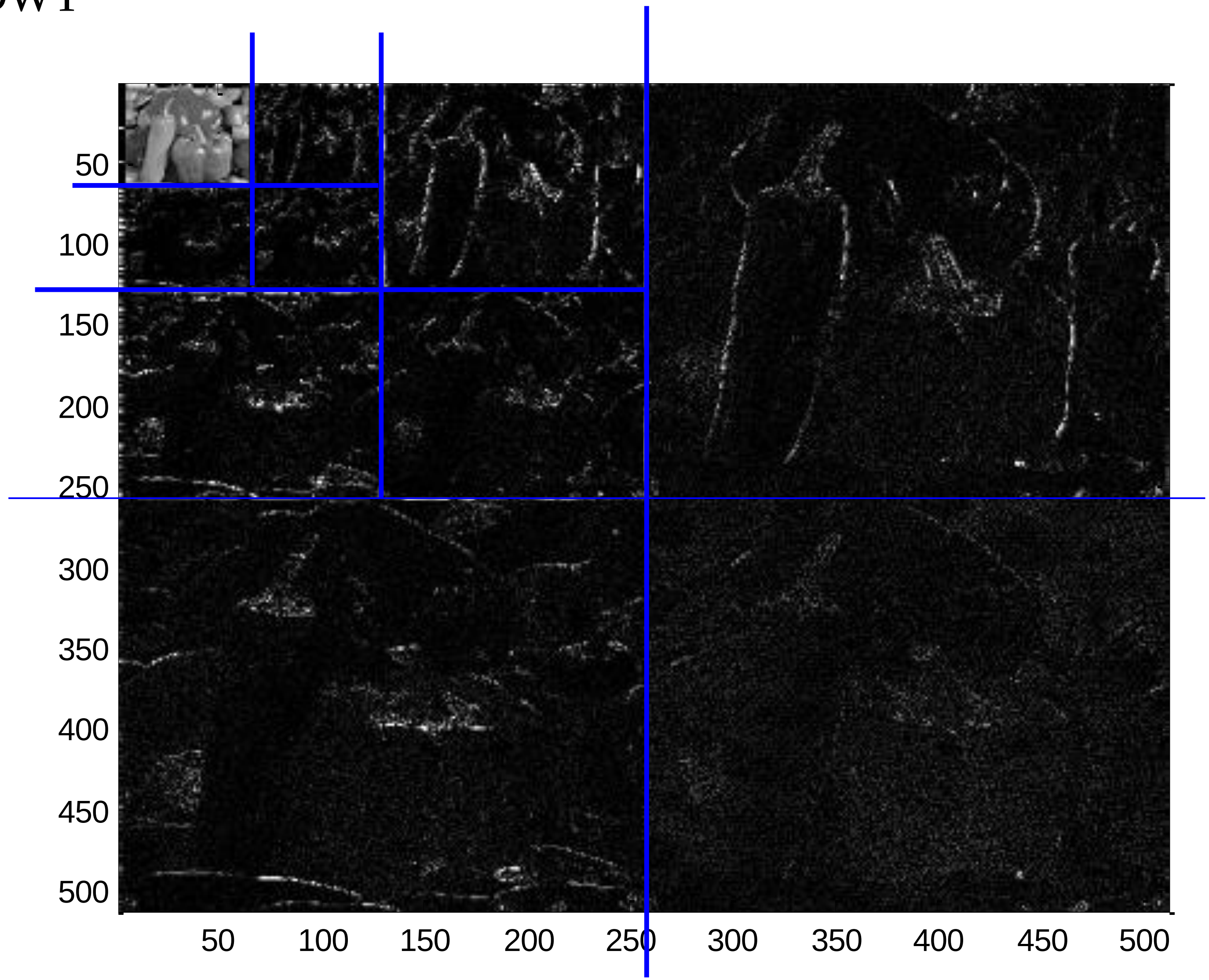
的結果

$x_{1,L}[m, n]$ $x_{1,H2}[m, n]$




$x_{1,H1}[m, n]$ $x_{1,H3}[m, n]$

3次2-D DWT
的結果


50
100
150
200
250
300
350
400
450
500
50
100
150
200
250
300
350
400
450
500

應用： 影像壓縮 (JPEG 2000)

其他應用：edge detection

corner detection

filter design

pattern recognition

music signal processing

economical data

temperature analysis

feature extraction

biomedical signal processing

# 附錄一：聲音檔的處理 (by Matlab)

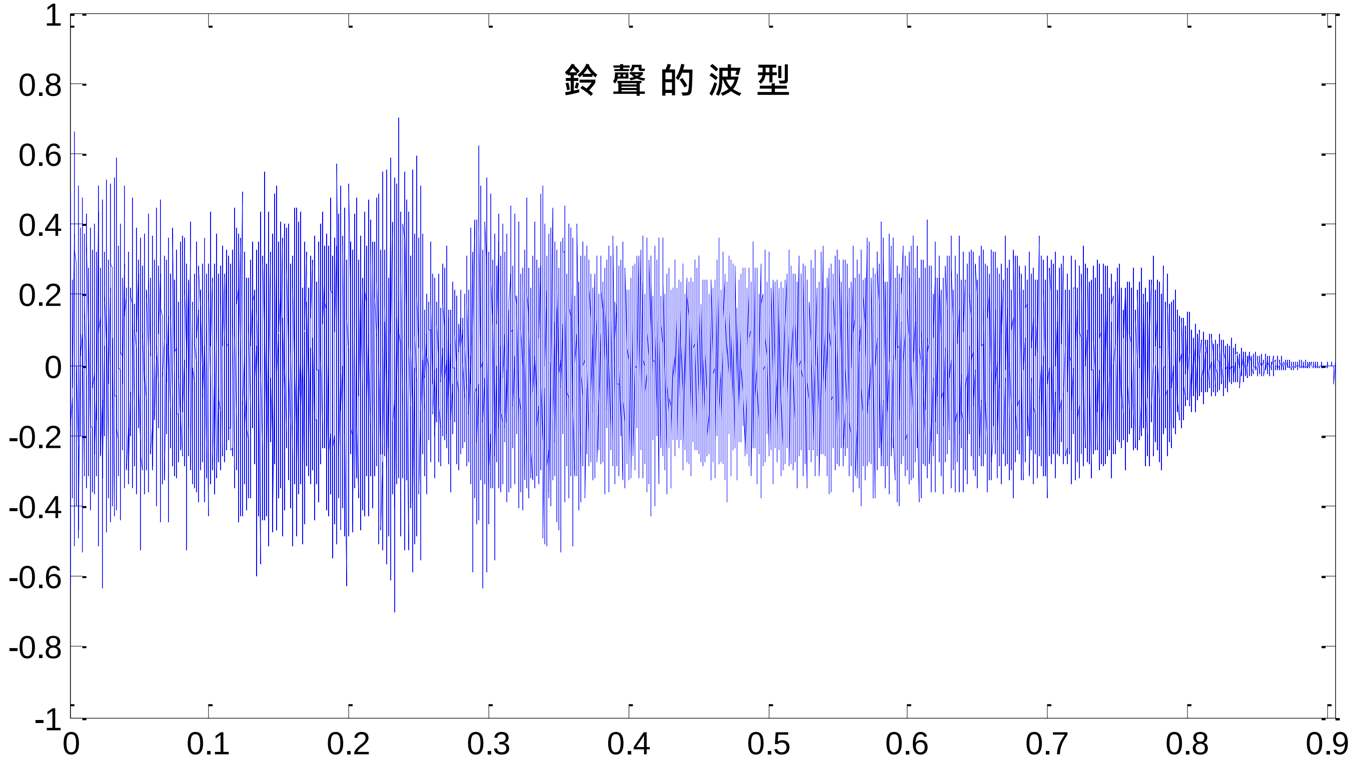
## A. 讀取聲音檔

- 電腦中，沒有經過壓縮的聲音檔都是 *.wav 的型態
- 讀取： **wavread** (2015以後的版本改成 **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)**




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 : $2^{-2}$， ….. ，

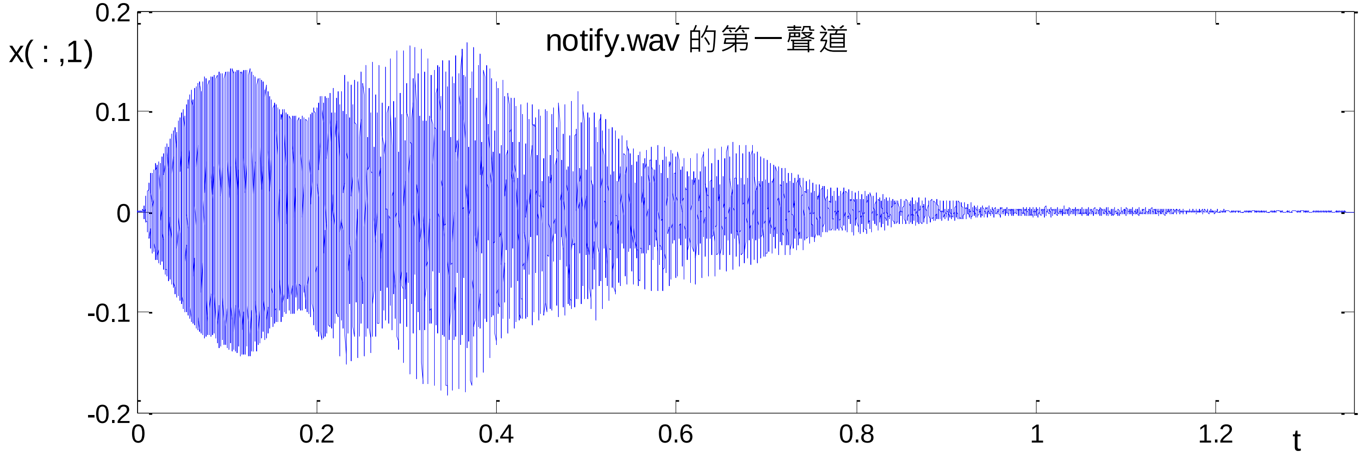
第 n 個bit : $2^{-nbits+1}$， 所以 **x** 乘上$2^{nbits-1}$ 是一個整數

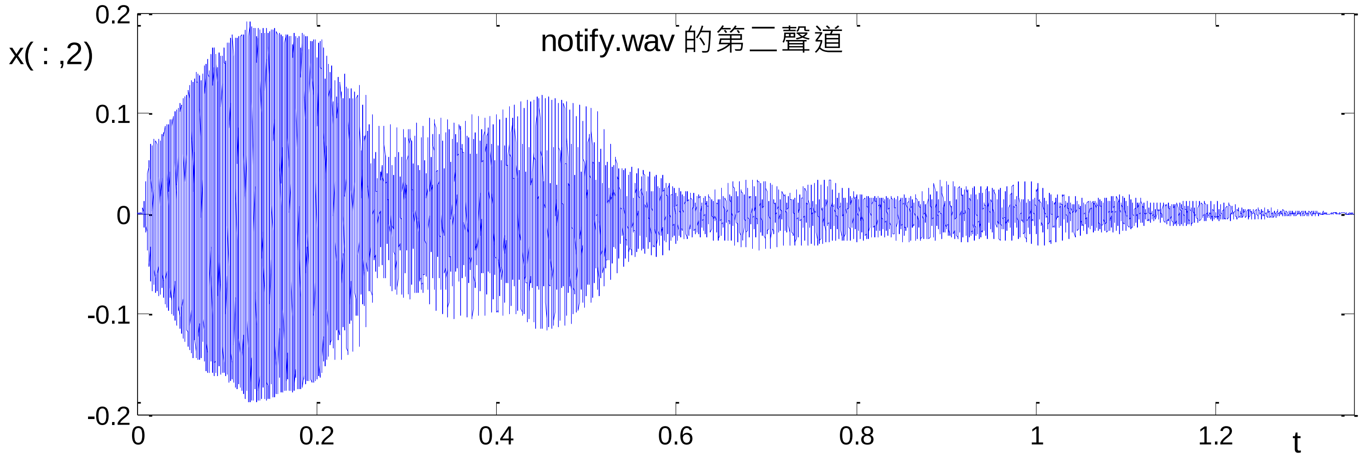
以鈴聲的例子， nbits = 8，所以 **x** 乘上 128是個整數

- 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');

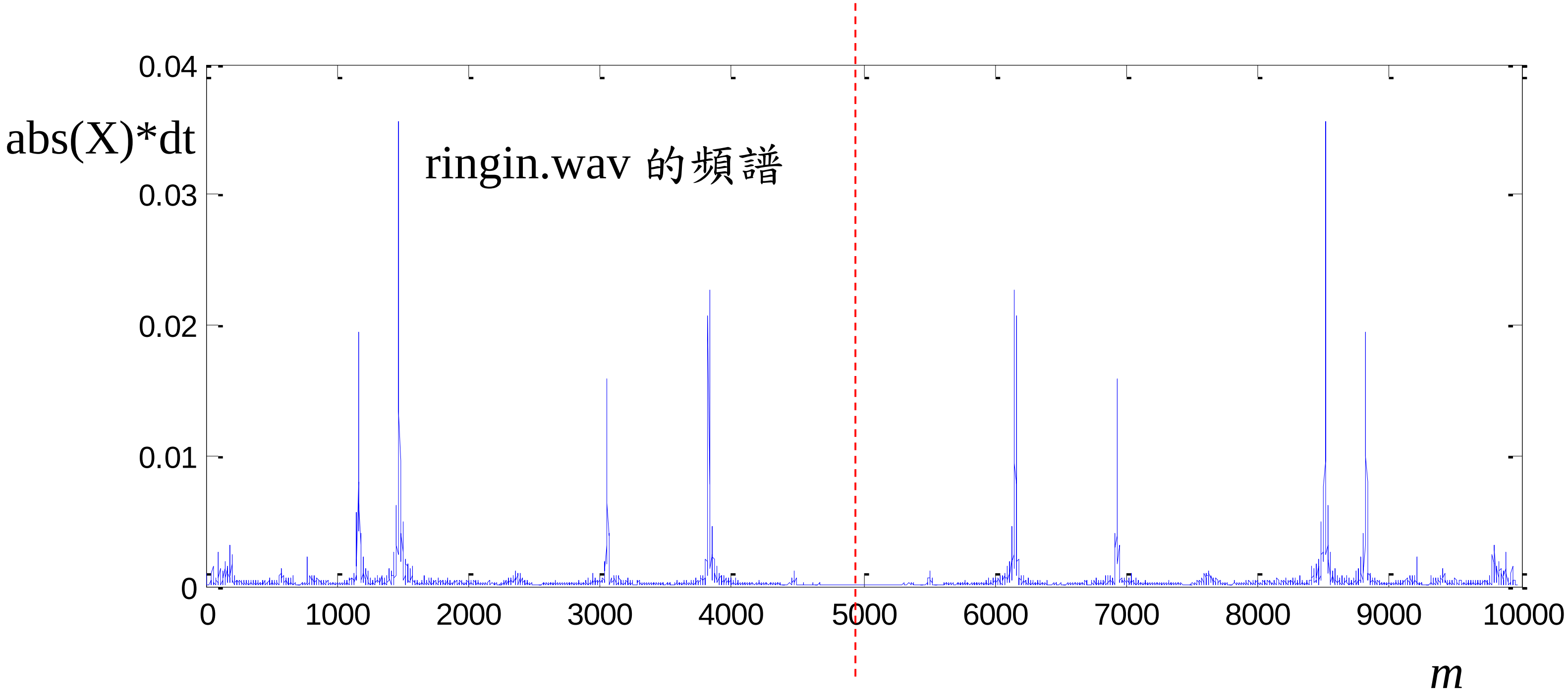
size(x) = 29823    2          fs = 22050



## B. 繪出頻譜

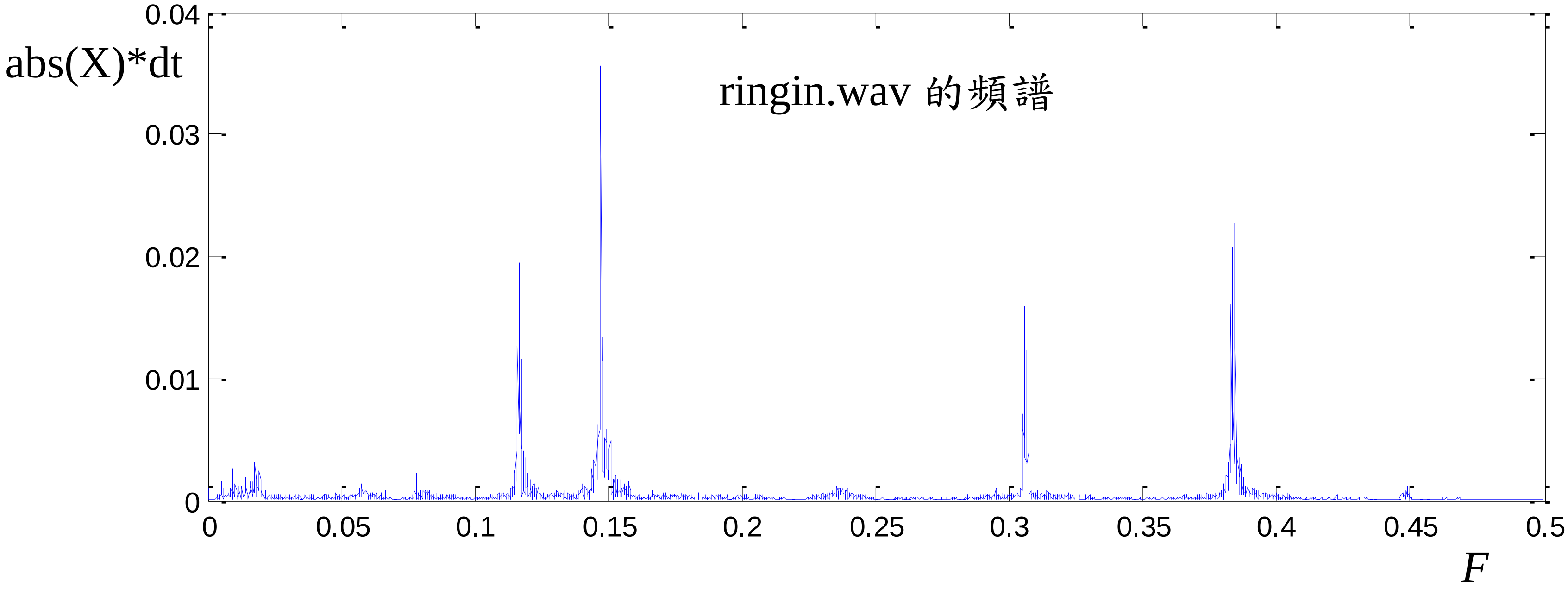
**X = fft(x);**　　　plot(abs(X)*dt);　% dt = 1/fs

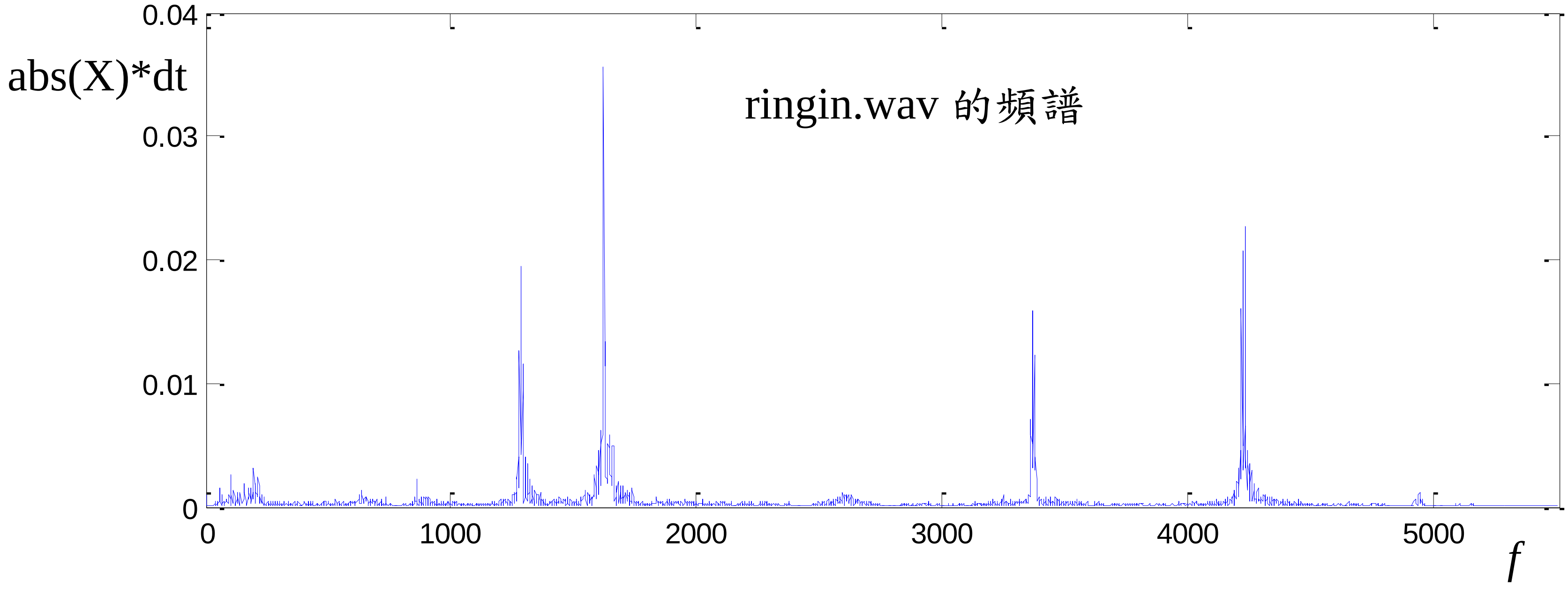



**fft 橫軸 轉換的方法**

(1) Using normalized frequency $F$:　　$F = m / N$.

(2) Using frequency $f$,　$f = F \times f_s = m \times (f_s / N)$.


abs(X)*dt
ringin.wav 的頻譜
0.04
0.03
0.02
0.01
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
F


abs(X)*dt
ringin.wav 的頻譜
0.04
0.03
0.02
0.01
0
0
1000
2000
3000
4000
5000
f

## C. 聲音的播放

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

(時間間隔 = 1/11025 = $9.07 \times 10^{-5}$ 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

(2015以後的版本改成 **audiowrite**)

## 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)
```