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！

上課方式

(1) 錄影 ，影片將藉由 NTU Cool 下載　http://cool.ntu.edu.tw

(2) 現場 (明達館231室)

作業和報告繳交方式

用 NTU Cool 來繳交作業與報告的電子檔　http://cool.ntu.edu.tw

注意，Tutorial 一定要交 Word 或 Latex 原始碼
Wiki 要寄編輯條目的連結給老師

- **評分方式：**

平時分數: 15 scores

基本分 12 分，各位同學皆可拿到

另外，每週都將會在課堂上問一個問題，請特定學號尾數的同學，在作業上回答 (每位同學每學期會輪到四次左右)，每正確回答一次加 0.8 分

將來再視疫情決定是否要在上課時問答

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

方式有五種，可任選其中一種

(1) 書面報告

1 column

(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。

格式不限，但儘量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。鼓勵多做實驗及模擬。

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

(2) Tutorial

pages 12~13

限十五個名額，和書面報告格式相同，但 17頁以上(若為加強前人的tutorial，則頁數為 $(2/3)N + 12$ 以上，N 為前人 tutorial 之頁數)，題目由老師指定 (see pages 12 and 13)，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。

or Latex 原始碼

選擇這個項目的同學，學期成績加4分

(3) 口頭報告

限~~五~~十個名額，每個人 30分鐘，題目可選擇和課程有關的任何一個主題。口頭報告的同學，請在 12月19日之前將報告的影片寄給老師。有意願的同學，請儘早告知，以先登記的同學為優先。

口頭報告鼓勵同學們參與聽講和票選，有參與票選的同學期末報告成績加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12月12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

編輯完成之後，要將連結寄給老師

(1)~(4) 各項期末報告若有做實驗模擬，請附上程式碼，會有額外的加分
(鼓勵不強制)

書面報告、Tutorial、編輯 Wikipedia、和程式編寫的期限是 12月26日

上課時間：14 週

9/5,

9/12, 出 HW1

9/19,

9/26, 交 HW1

10/3, 出 HW2

10/17,

10/24, 交 HW2

3n-1 出 homework

3n+1 交 homework

10/31, 出 HW3

11/7,

11/14, 交 HW3

11/21, 出 HW4

11/28,

12/5, 交 HW4,

12/12, 出 HW5

12/19, 口頭報告的同學交影片

12/26, 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) Applications of Time-Frequency Analysis

(續)

time-frequency analysis

課程大綱：

(10) Hilbert Huang Transform

(11) From Haar Transforms to Wavelet Transforms

(12) Continuous Wavelet Transforms with Discrete Coefficients

(13) Discrete Wavelet Transform

(14) Applications of the Wavelet Transform

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

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

(4) P. Flandrin, *Time-frequency / Time Scale Analysis*, translated by J. Stöckler, Academic Press, San Diego, 1999.

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

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

(7) F. Hlawatsch and F. Auger, *Time-frequency Analysis Concepts and Methods*, Wiley, London, 2008.

Matlab Program

Download: 請洽台大各系所

參考書目

洪維恩，Matlab 程式設計，旗標，台北市，2013 . (合適的入門書)

張智星，Matlab 程式設計入門篇，第四版，碁峰，2016.

預計看書學習所花時間： 3~5 天

Python Program

Download: https://www.python.org/

參考書目

葉難， Python程式設計入門，博碩，2015

黃健庭， Python程式設計：從入門到進階應用 ，全華，2020

The Python Tutorial   https://docs.python.org/3/tutorial/index.html

Tutorial 可供選擇的題目(可以略做修改)

(1) Wiener filter in the Time Frequency Plane

✓(2) Prediction Techniques Using Time-Frequency Analysis

(3) Log-Mel Spectrogram

(4) Local Polynomial Fourier Transform

(5) Recent Development of Compressive Sensing

(6) Fundamental Frequency Estimation Techniques

(7) Sideband Analysis

(8) Frequency Modulated Continuous Wave (FMCW) Radar chirp modulation

✓(9) Linear Canonical Transform with Complex Parameters

✓(10) Ensemble Empirical Mode Decomposition (EEMD)

Tutorial 可供選擇的題目(可以略做修改)

(11) Variational Mode Decomposition (VMD)

[✓ (12) Legendre Wavelet and Hermite Wavelet]

(13) Empirical Wavelet Transform

[✓ (14) Wavelet Transforms for Face Detection]

(15) Wavelet Transform for Video Analysis

I. Introduction

Fourier transform (FT)

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

↑ *t* varies from –∞~∞

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

Cosine Transform, Sine Transform, Z Transform.

Some things make the FT not practical:

(1) It less happens that a signal has the interval of (–∞~∞)

Even for a signal with infinite length, we are only interested in the characteristics in a finite interval.

(2) It is hard to observe the variation of spectrum with time by the FT.

Do : 261.63 Hz $261.63 \times 2^{k/12}$

La : $261.63 \times 2^{-3/12}$ Mi : $261.63 \times 2^{4/12}$

Example 1: $x(t) = \cos(440\pi t)$ when $t < 0.5$, ±220Hz 低 La

$x(t) = \cos(660\pi t)$ when $0.5 \le t < 1$, ±330Hz Mi

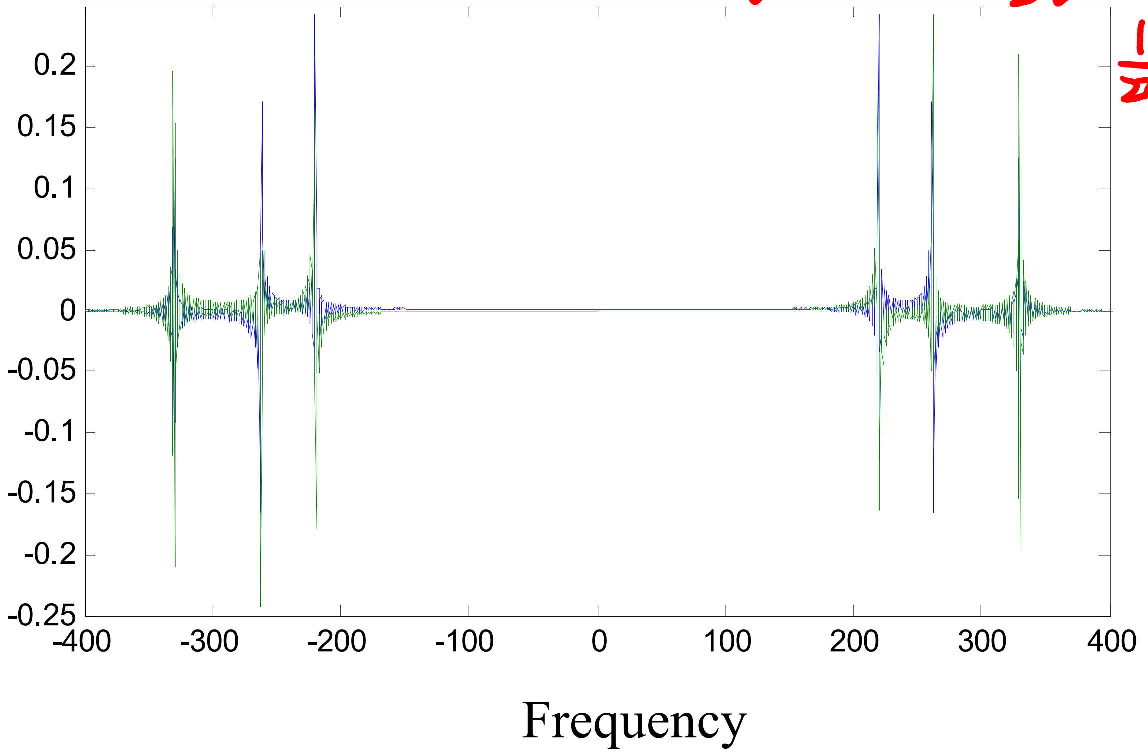
$x(t) = \cos(524\pi t)$ when $t \ge 1$ ±262Hz Do

$\cos(440\pi t) = (\exp(j440\pi t) + \exp(-j440\pi t))/2$

frequency $= \frac{1}{2\pi}\frac{d}{dt}(440\pi t) = \frac{440\pi}{2\pi} = 220$,

$\frac{1}{2\pi}\frac{d}{dt}(-440\pi t) = -220$

The Fourier transform of $x(t)$



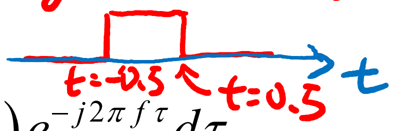
(A) Finite-Supporting Fourier Transform

ex: $t_0 = 0.7$, $B = 0.1$
$0.6 \sim 0.8$

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

$\Pi(t)$: rectangular function

$\Pi(t)$

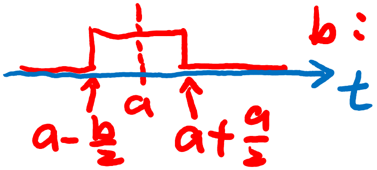


$$\Rightarrow X(f) = \int_{t-B}^{t+B} x(\tau) e^{-j2\pi f \tau} d\tau = \int_{-\infty}^{\infty} \Pi\left(\frac{\tau - t}{2B}\right) x(\tau) e^{-j2\pi f \tau} d\tau$$

a: center
b: width

(B) Short-Time Fourier Transform (STFT)

$\Pi\left(\frac{t-a}{b}\right)$



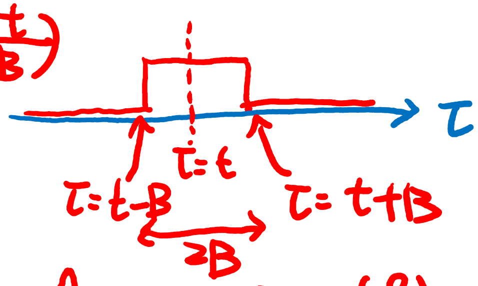
$$X(t, f) = \int_{-\infty}^{\infty} w(t - \tau) x(\tau) e^{-j2\pi f \tau} d\tau$$

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

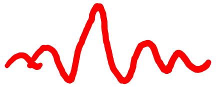
STFT 也稱作 windowed Fourier transform 或

time-dependent Fourier transform

$\Pi\left(\frac{\tau-t}{2B}\right)$

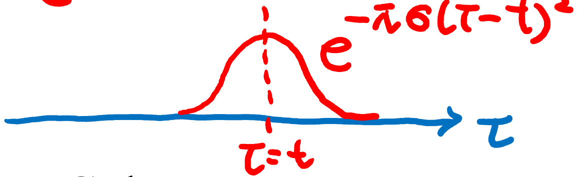


$\Pi(t) \xrightarrow{FT}$



$\text{sinc}(f)$
$= \sin(\pi f)/\pi f$

$w(t) = e^{-\pi\sigma t^2}$
$w(t-\tau) = e^{-\pi\sigma(\tau-t)^2}$



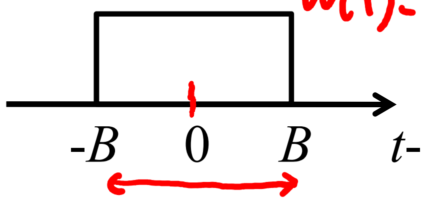
[Ref] L. Cohen, *Time-Frequency Analysis*, Prentice-Hall, New York, 1995.

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

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

最簡單的例子： $w(t) = 1$ for $|t| \le B$,

$w(t) = 0$ otherwise



$w(t)=\Pi\left(\frac{t-0}{2B}\right)=\Pi\left(\frac{t}{2B}\right)$

$w(t-\tau)=\Pi\left(\frac{t-\tau}{2B}\right)$

$(\because \Pi(t)=\Pi(-t)) = \Pi\left(\frac{\tau-t}{2B}\right)$

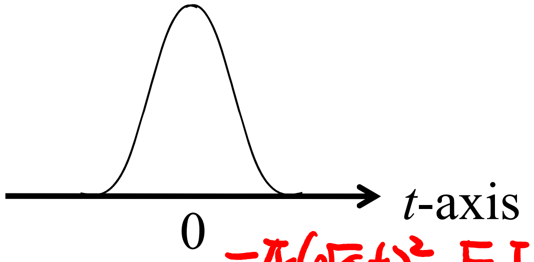
此時 Short-time Fourier transform 可以改寫

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

σ : sigma

其他的例子：

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



$e^{-\pi t^2} \xrightarrow{FT} e^{-\pi f^2}$ from page30 , $e^{-\pi(\sqrt{\sigma}t)^2} \xrightarrow{FT} \frac{1}{\sqrt{\sigma}} e^{-\pi\left(\frac{f}{\sqrt{\sigma}}\right)^2} = \frac{1}{\sqrt{\sigma}} e^{-\pi\frac{f^2}{\sigma}}$

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

此時的 Short-Time Fourier Transform 亦稱作 Gabor Transform

(C) Gabor Transform

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

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

- S. Qian and D. Chen, *Joint Time-Frequency Analysis: Methods and Applications*, Prentice Hall, N.J., 1996.

Without cross term, poor clarity

(D) Wigner Distribution Function

$$G_x\left(t,f\right)=\int_{-\infty}^{\infty}e^{-j2\pi f\tau}x\left(t+\tau/2\right)x^*\left(t-\tau/2\right)d\tau$$

$$G_x\left(t,\omega\right)=\int_{-\infty}^{\infty}e^{-j\omega\tau}x\left(t+\tau/2\right)x^*\left(t-\tau/2\right)d\tau$$

With cross term, high 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 $t \ge 1$

number of sampling points $= 660 \times 1.5 = \boxed{990}$, $\Delta_t < \frac{1}{660}$

adaptive sampling

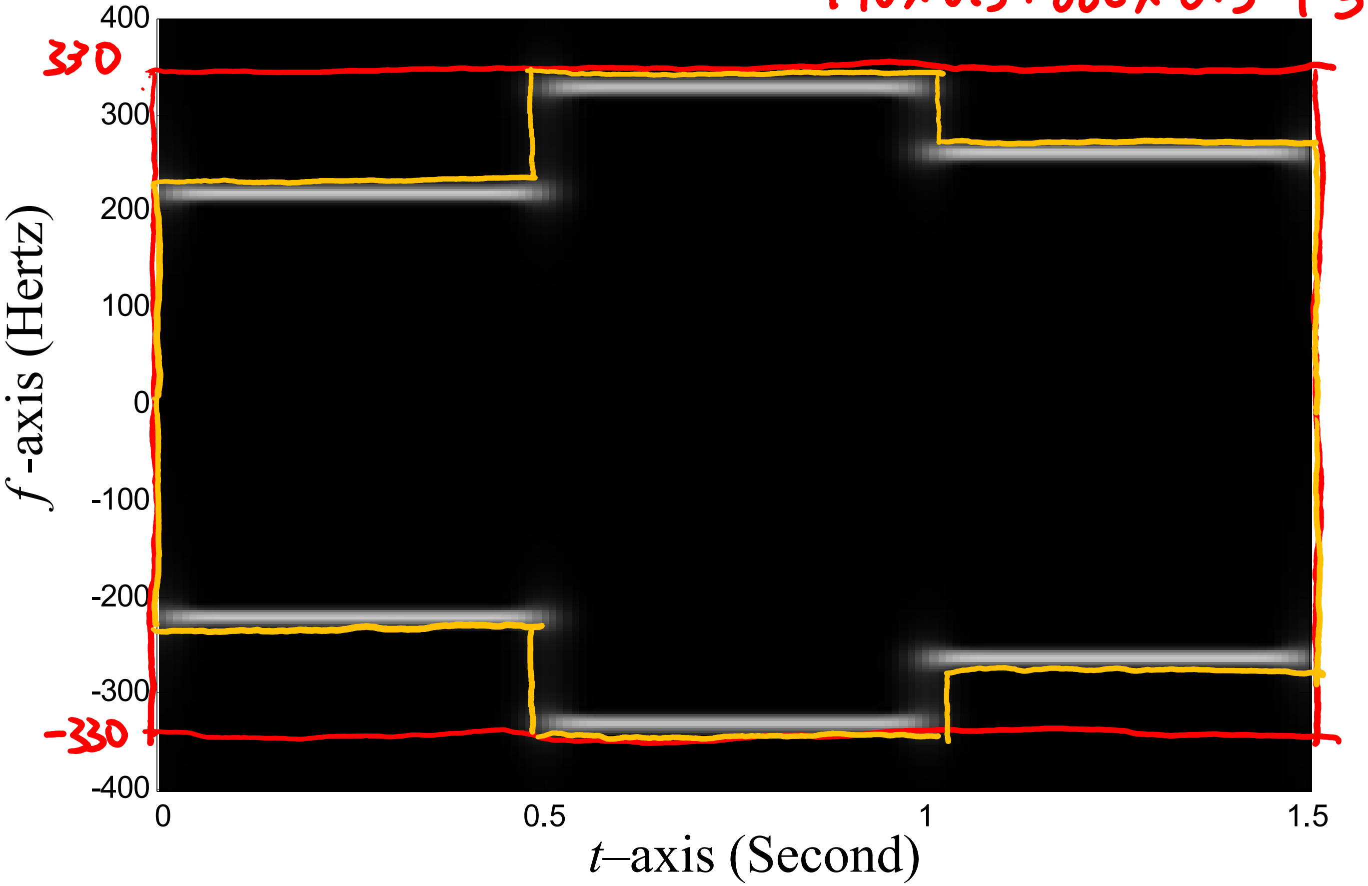
$\Delta_t < \frac{1}{440}$; $\Delta_t < \frac{1}{660}$, $\Delta_t < \frac{1}{524}$

for $0 < t < 0.5$ $0.5 < t < 1$ $1 < t < 1.5$

number of sampling points

$440 \times 0.5 + 660 \times 0.5 + 524 \times 0.5 = \boxed{812}$

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



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

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)

$f =$

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

$\omega =$

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

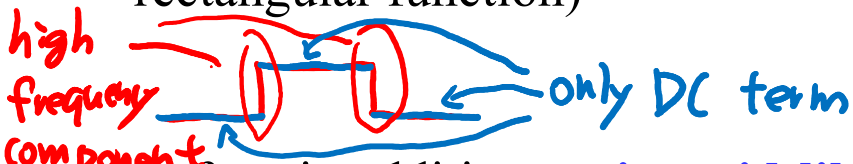
Speech

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

lens : 乘 chirp

free space : convolution with chirp

地震波



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

$$\sum_n a_n \cos(2\pi f_n t + \phi_n)$$

for $-\infty < t < \infty$

- Sinusoid Function

- Chirp function 啁啾聲

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

$\phi(t)=\alpha_2 t^2+\alpha_1 t+\alpha_0$ $\quad\frac{1}{2\pi}\frac{d}{dt}\phi(t)=\frac{1}{2\pi}(2\alpha_2 t+\alpha_1)=\frac{\alpha_2}{\pi}t+\frac{\alpha_1}{2\pi}$

acoustics, wireless communication, radar system, optics

Example：the 2nd, 3rd, 4th, and 5th tones

- Higher order exponential function

order ≥ 2

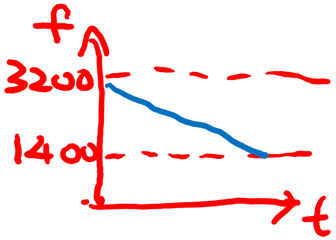
Example 2

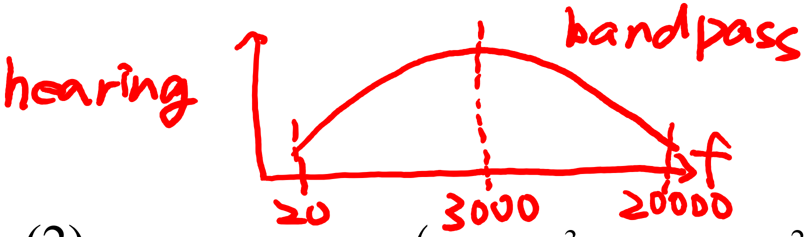
also a chirp

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

$=0.25(\exp(j\phi(t))+\exp(-j\phi(t)))$ $\phi(t)=6400\pi t-600\pi t^2$

$\frac{1}{2\pi}\frac{d}{dt}(\pm\phi(t))=\pm(3200-600t)$





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

$\phi(t)=600\pi t^3-2700\pi t^2+5050\pi t$

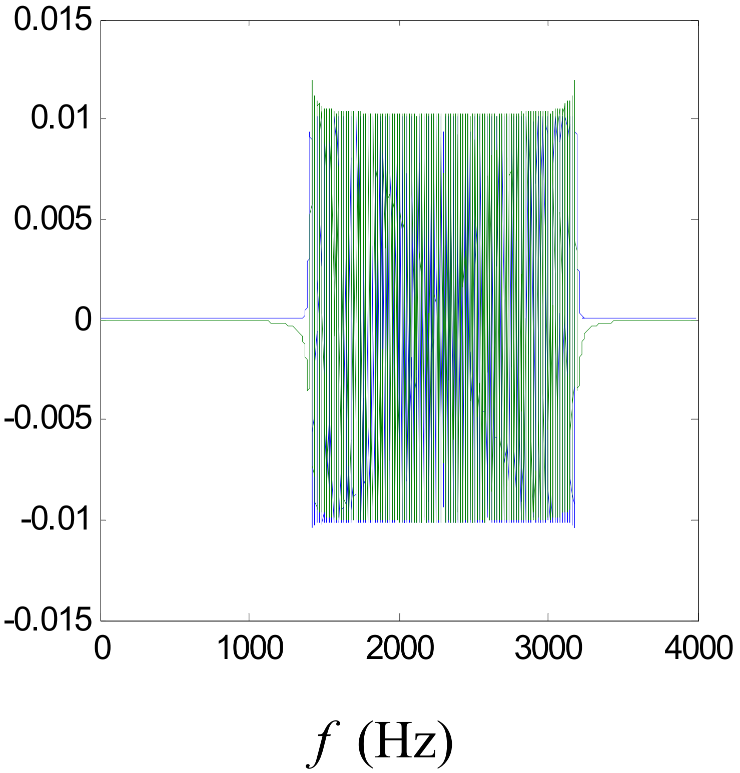
$\frac{1}{2\pi}\frac{d}{dt}(\pm\phi(t))=\pm(900t^2-2700t+2525)$

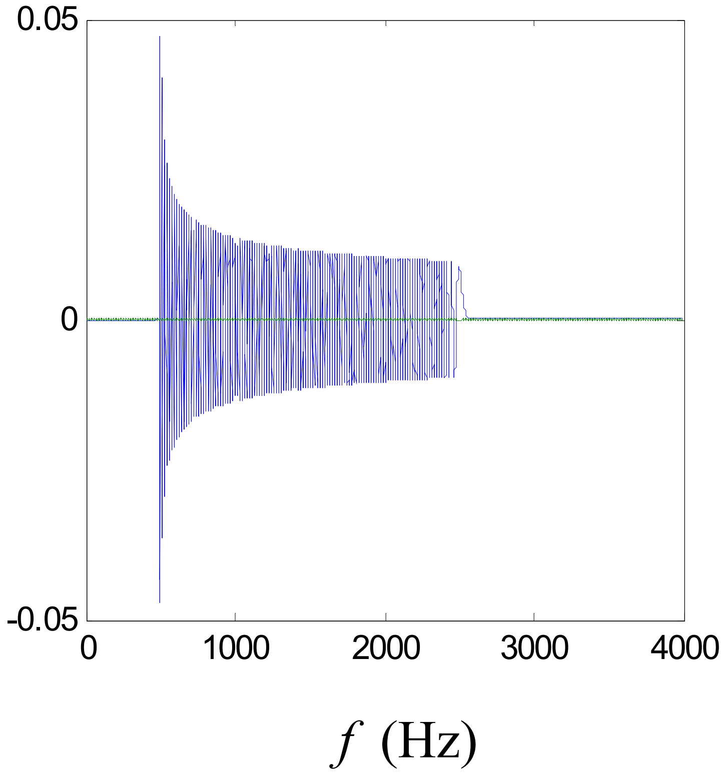
$=\pm(900(t-1.5)^2+500)$

Fourier transform

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

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





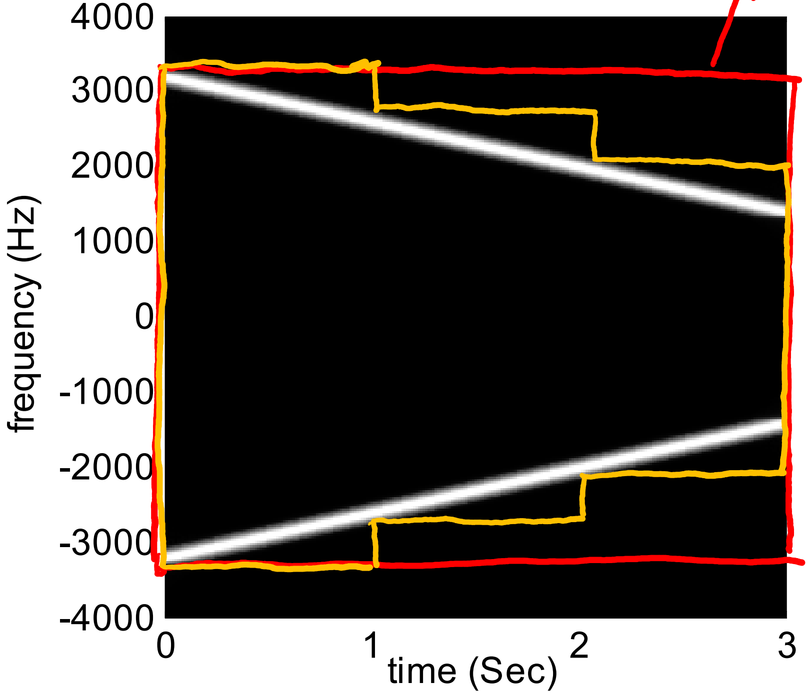
(1)

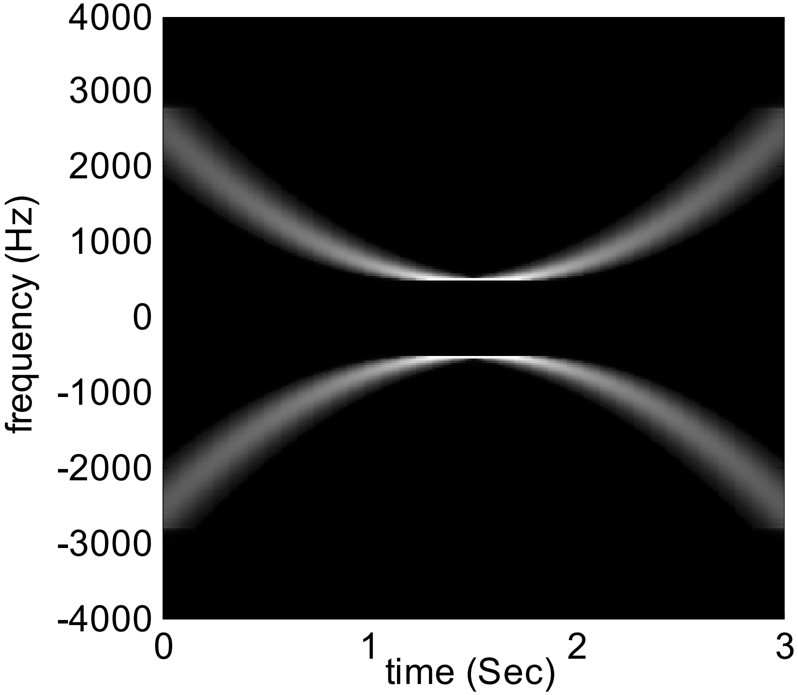
$$x(t) = 0.5\cos\left(6400\pi t - 600\pi t^2\right)$$

(2)

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

area=19200





original sampling points

$\Delta_t < \frac{1}{6400}$

$6400 \times 3 = 19200$

If Δ_t changes for every 1 second

$\Delta_t < \frac{1}{6400}$, $\Delta_t < \frac{1}{5200}$, $\Delta_t < \frac{1}{4000}$

$0 < t < 1$ $1 < t < 2$ $2 < t < 3$

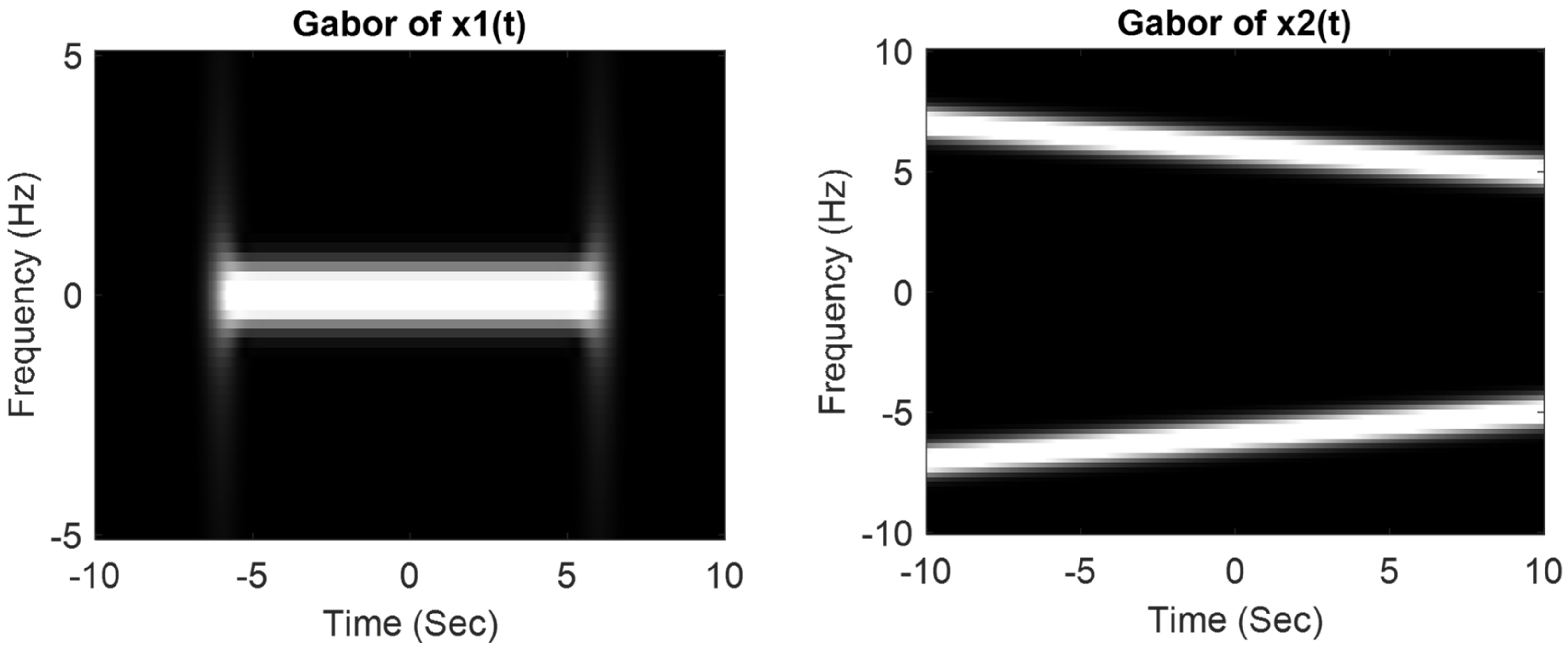
number of sampling points = ?

Example 3

$\Pi(\frac{t}{12})$

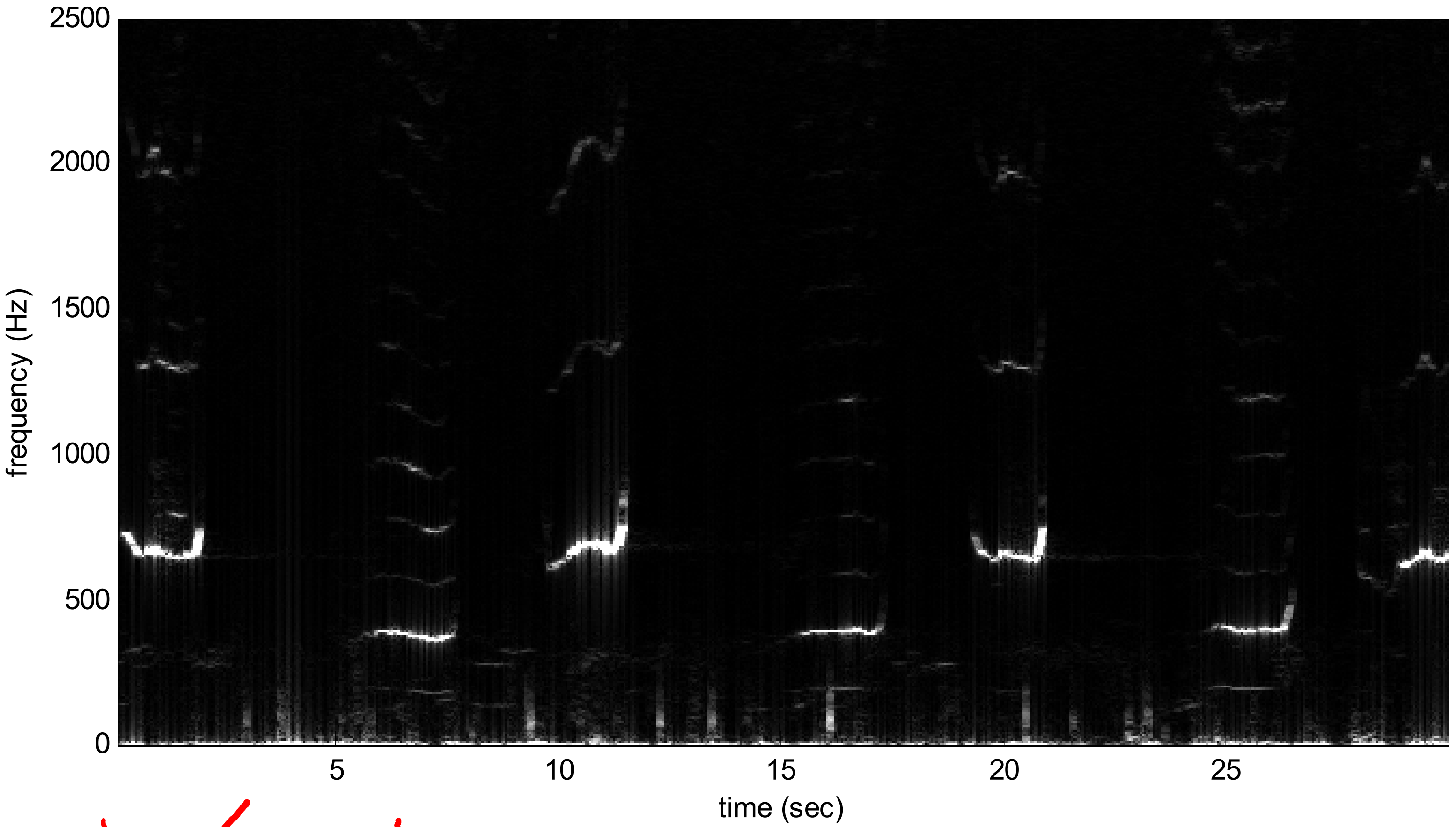
left: $x_1(t) = 1$ for $|t| \le 6$, $x_1(t) = 0$ otherwise, right: $\underline{x_2(t)} = \underline{\cos(12\pi t - 0.1\pi} \underline{t^2)}$

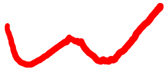
Gabor transform



Example 4

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





3 extrem·s

order of $\phi(t) \geq \sim$

Why Time-Frequency Analysis is Important?

applications of the Fourier transform — spectrum analysis
computing convolution

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

spectrum analysis may be replaced by time-frequency analysis

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

- modulation,
- multiplexing,
- filter design,
- data compression,
- signal analysis,

time-frequency 10^{-1}

Fourier 10^{-3}

- signal identification,
- acoustics,
- system modeling,
- radar system analysis
- sampling

$\Delta_t < \frac{1}{2B}$

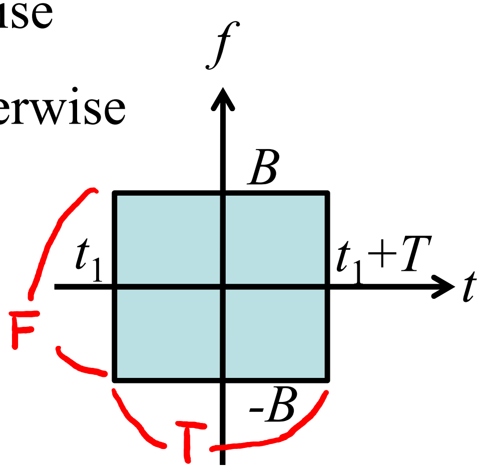
B: bandwidth

Example: Generalization for sampling theory

假設有一個信號，

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

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



根據取樣定理， $\underline{\Delta_t \leq 1/F}$, $F = 2B$, B:頻寬

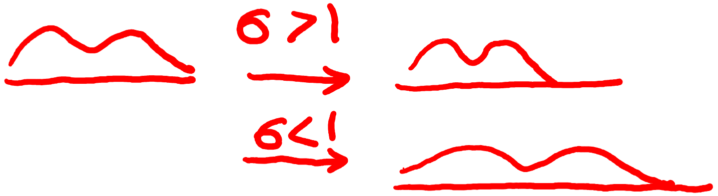
所以，取樣點數 N 的範圍是

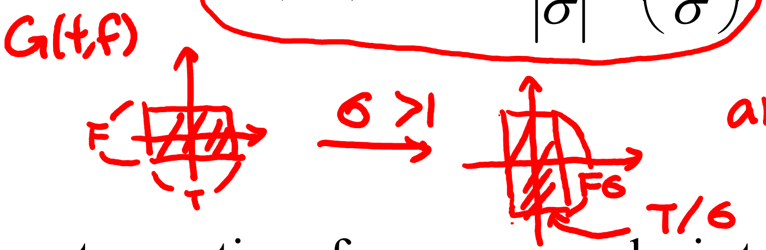
$$N = T/\Delta_t \geq TF$$

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:

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

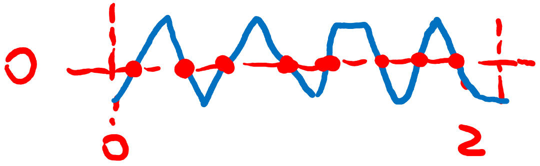




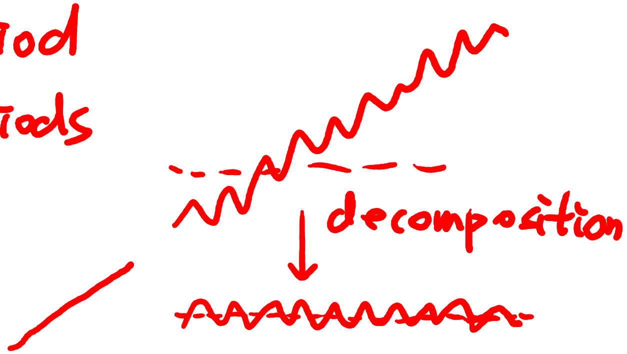
area are the same

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

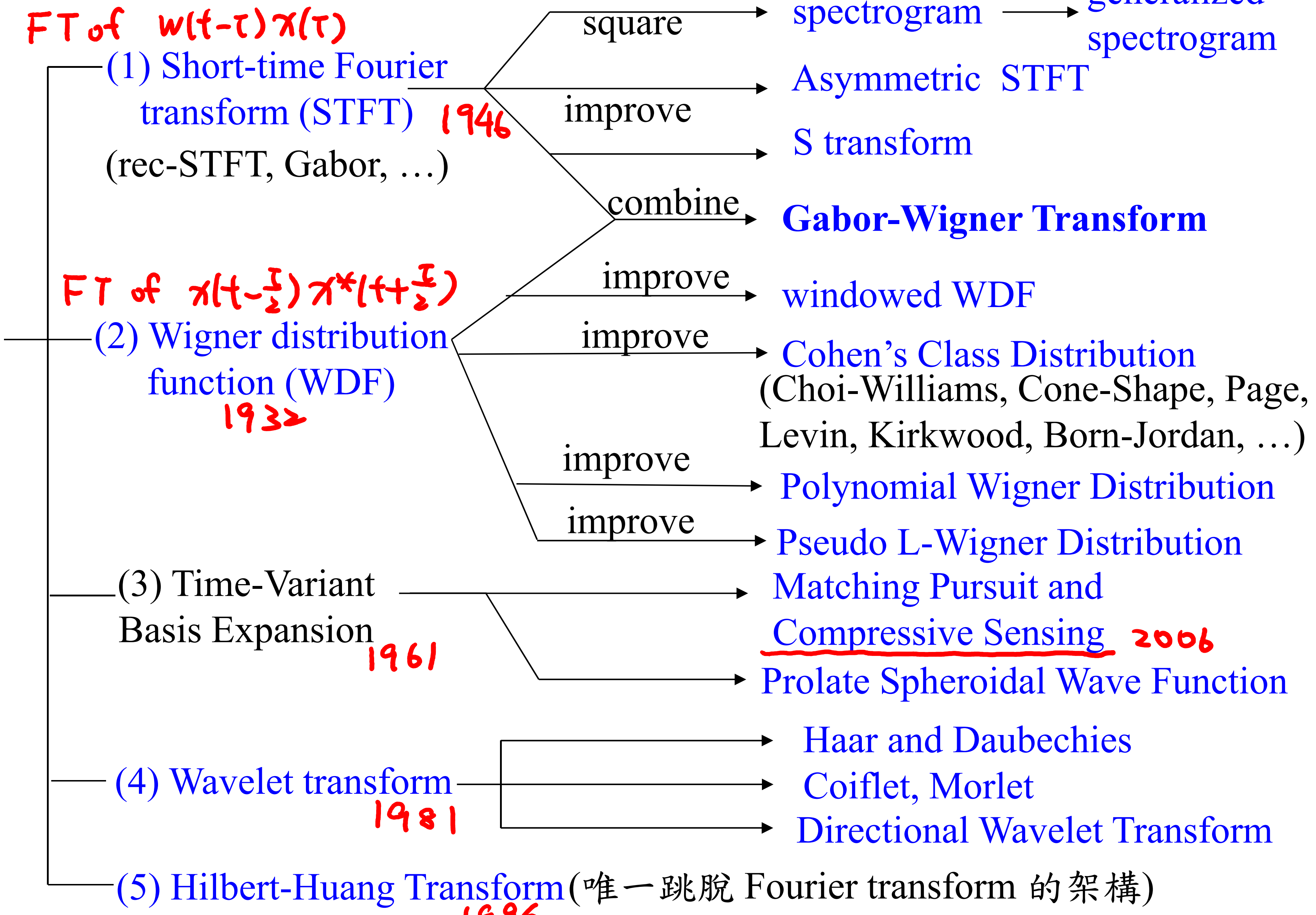
Time-frequency analysis is an efficient tool for adaptive signal processing.



2 zero crossing = 1 period
8 zero crossings = 4 periods
= 2 period/second
frequency = 2 Hz



時頻分析大家族



FT of $w(t-\tau)x(\tau)$

(1) Short-time Fourier transform (STFT) 1946

(rec-STFT, Gabor, …)

square → spectrogram → generalized spectrogram

improve → Asymmetric STFT

→ S transform

combine → **Gabor-Wigner Transform**

FT of $x(t-\frac{\tau}{2})x^*(t+\frac{\tau}{2})$

(2) Wigner distribution function (WDF) 1932

improve → windowed WDF

improve → Cohen's Class Distribution

(Choi-Williams, Cone-Shape, Page, Levin, Kirkwood, Born-Jordan, …)

improve → Polynomial Wigner Distribution

improve → Pseudo L-Wigner Distribution

(3) Time-Variant Basis Expansion 1961

→ Matching Pursuit and Compressive Sensing 2006

→ Prolate Spheroidal Wave Function

(4) Wavelet transform 1981

→ Haar and Daubechies

→ Coiflet, Morlet

→ Directional Wavelet Transform

(5) Hilbert-Huang Transform(唯一跳脫 Fourier transform 的架構) 1996

• Continuous Wavelet Transform

forward wavelet transform:

ψ: psy /saɪ/

ψ(t): some high frequency function with limited support

↓

範圍

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

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

$b>0$

inverse wavelet transform:

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

large b → small f

small b → high f

$\varphi_{a,b}(t)$ 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

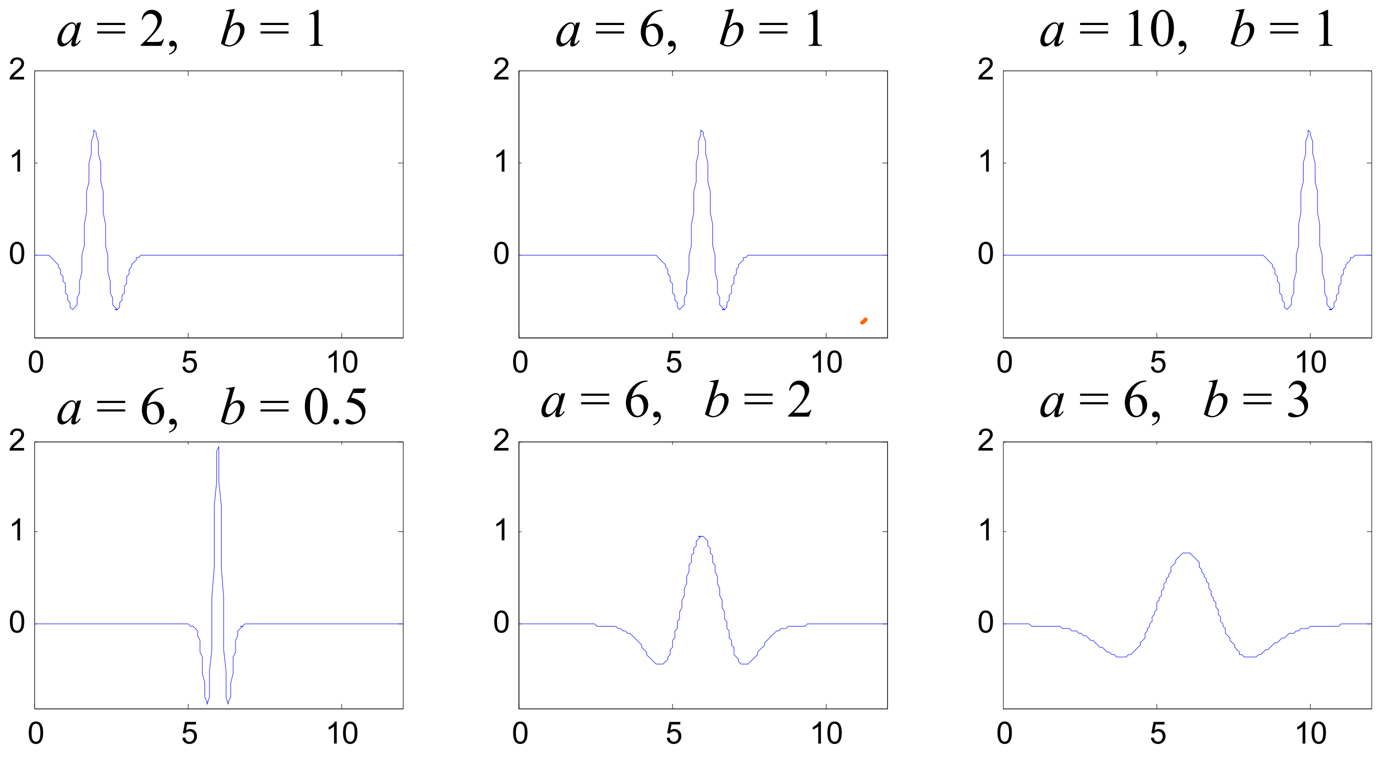
Mexican hat wavelet, Haar Wavelet, Daubechies wavelet, triangular wavelet, …………

$$\frac{d^2}{dt^2} e^{-\pi t^2} = (4\pi^2 t^2 - 2\pi) e^{-\pi t^2}$$

例子： Mexican hat wavelet (Laplacian of Gaussian) 隨 a and b 變化之情形

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

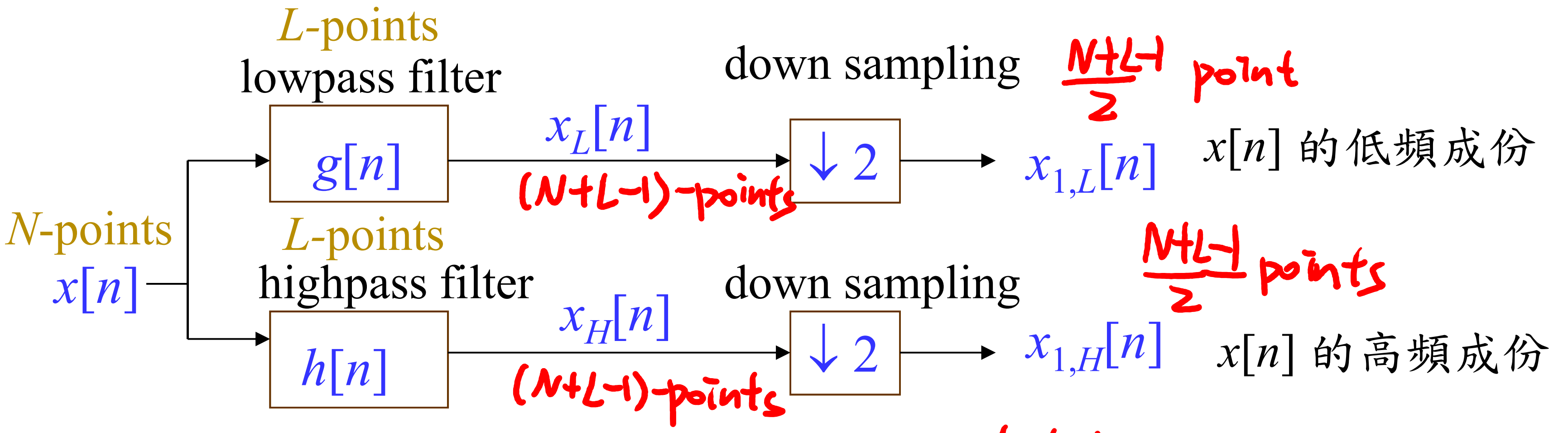
LoG



• Discrete Wavelet Transform (DWT)

application orient
1990

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



N-points

$x[n]$

L-points

lowpass filter

$g[n]$

$x_L[n]$

(N+L-1)-points

down sampling

$\downarrow 2$

$\frac{N+L-1}{2}$ point

$x_{1,L}[n]$

$x[n]$ 的低頻成份

L-points

highpass filter

$h[n]$

$x_H[n]$

(N+L-1)-points

down sampling

$\downarrow 2$

$\frac{N+L-1}{2}$ points

$x_{1,H}[n]$

$x[n]$ 的高頻成份

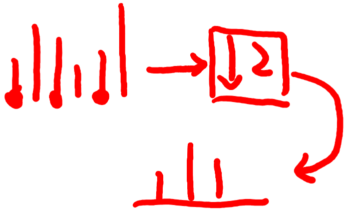
In usual, L<10

total: N+L-1 points
≅ N points

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

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

$$y[n] \longrightarrow \boxed{\downarrow Q} \longrightarrow y[Qn]$$



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

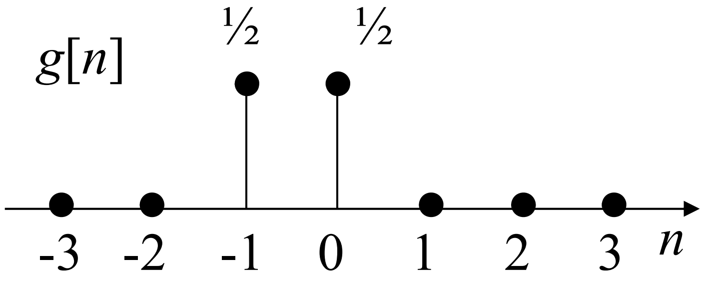
例子：2-point Haar wavelet

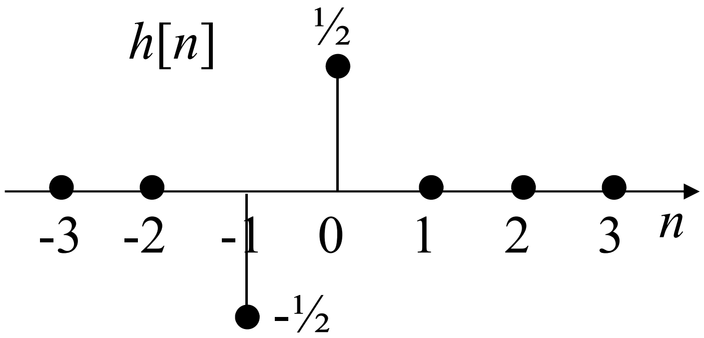
$g[n] = 1/2$ for $n = -1, 0$

$g[n] = 0$ 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）

Remember: For the 2-point DFT $X[m]=\sum_{k} e^{-j\frac{2\pi mn}{2}}x[n]=\sum_{k}(-1)^{mn}x[n]$

$$\begin{bmatrix} X[0] \\ X[1] \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \end{bmatrix}$$

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

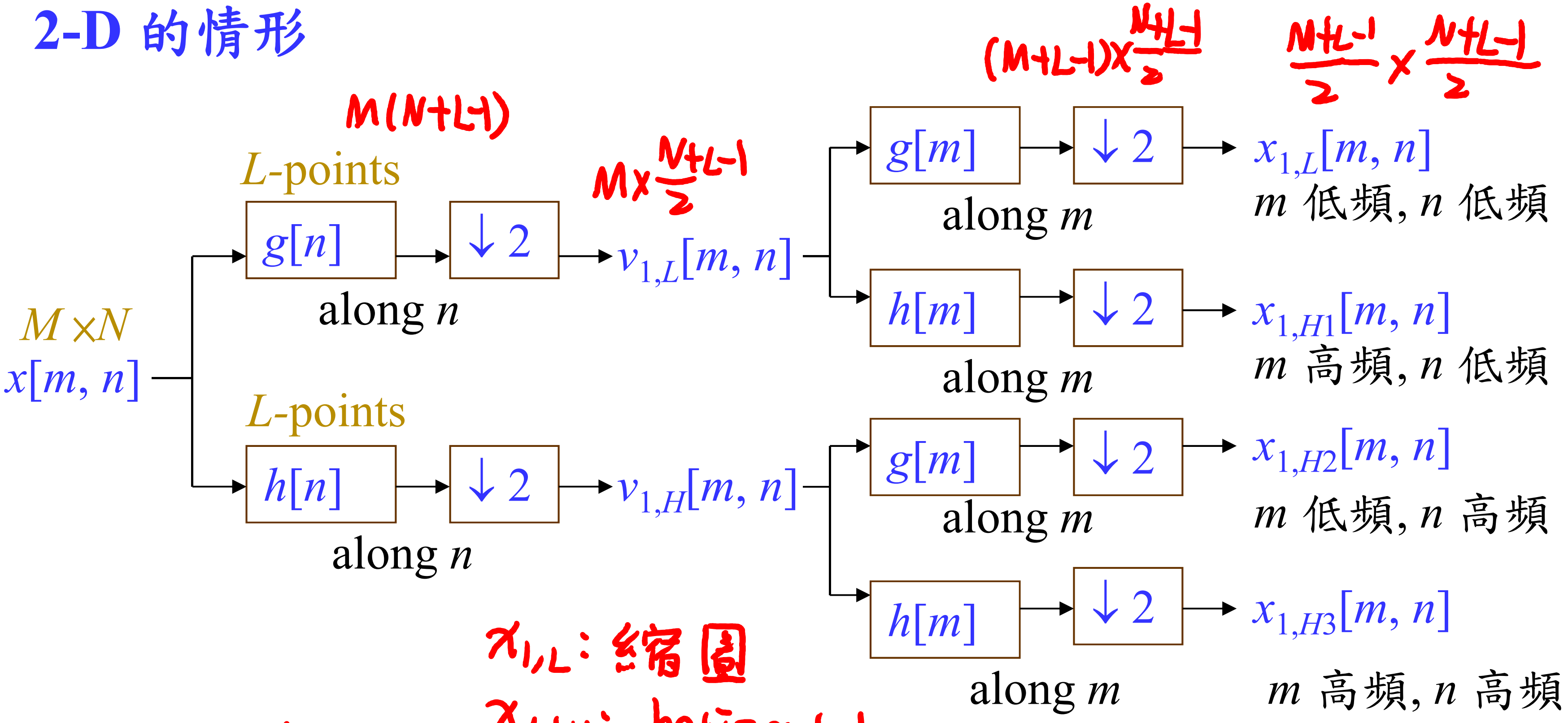
但是 $g[n]$ 通常都是 lowpass filter 的型態

$h[n]$ 通常都是 highpass filter 的型態

Discrete wavelet transform 有很多種

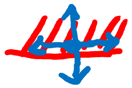
(discrete Haar wavelet, discrete Daubechies wavelet, B-spline DWT, symlet, coilet, ……..)

2-D 的情形



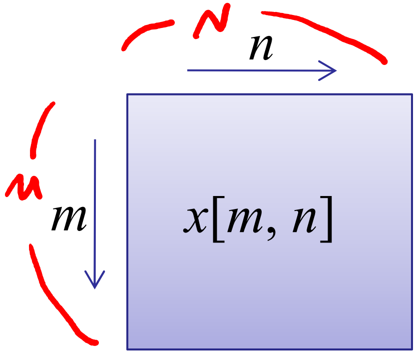
$x_{1,L}$: 縮圖

$x_{1,H1}$: horizontal edges

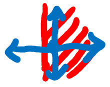


total size: $(M+L-1)(N+L-1)$

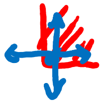
$\cong MN$



$x_{1,H2}$: vertical edges

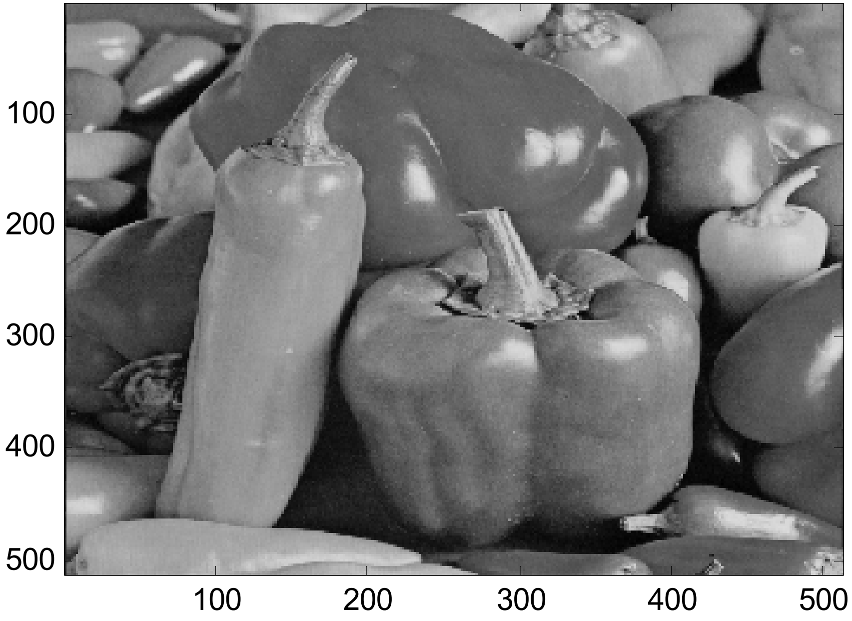


$x_{1,H3}$: corners or noise



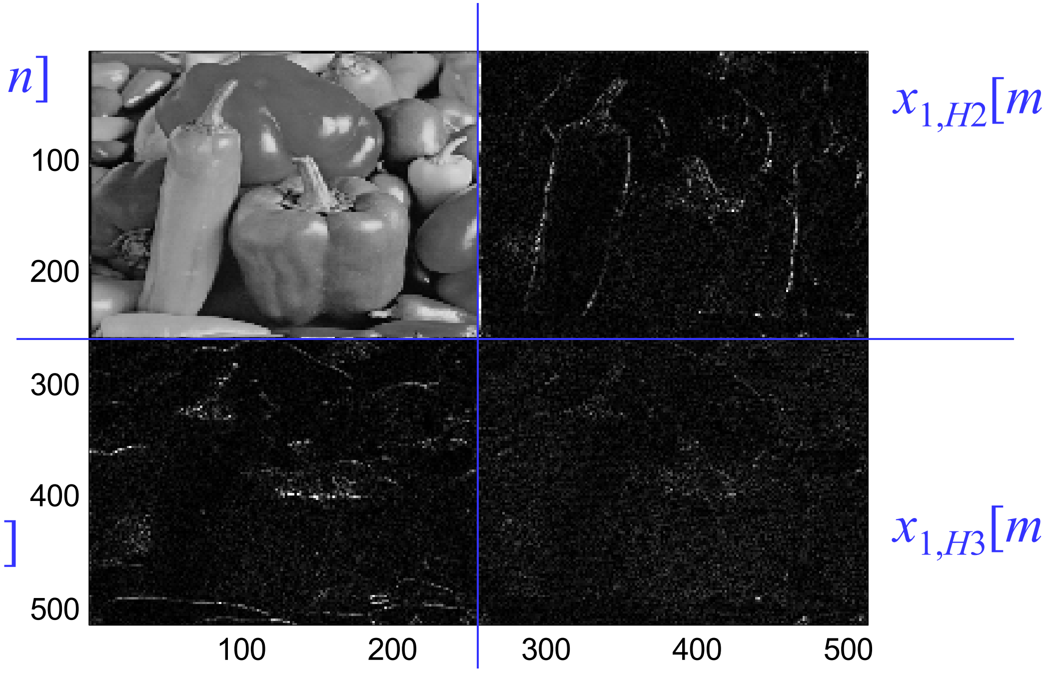
原影像

peppers.bmp

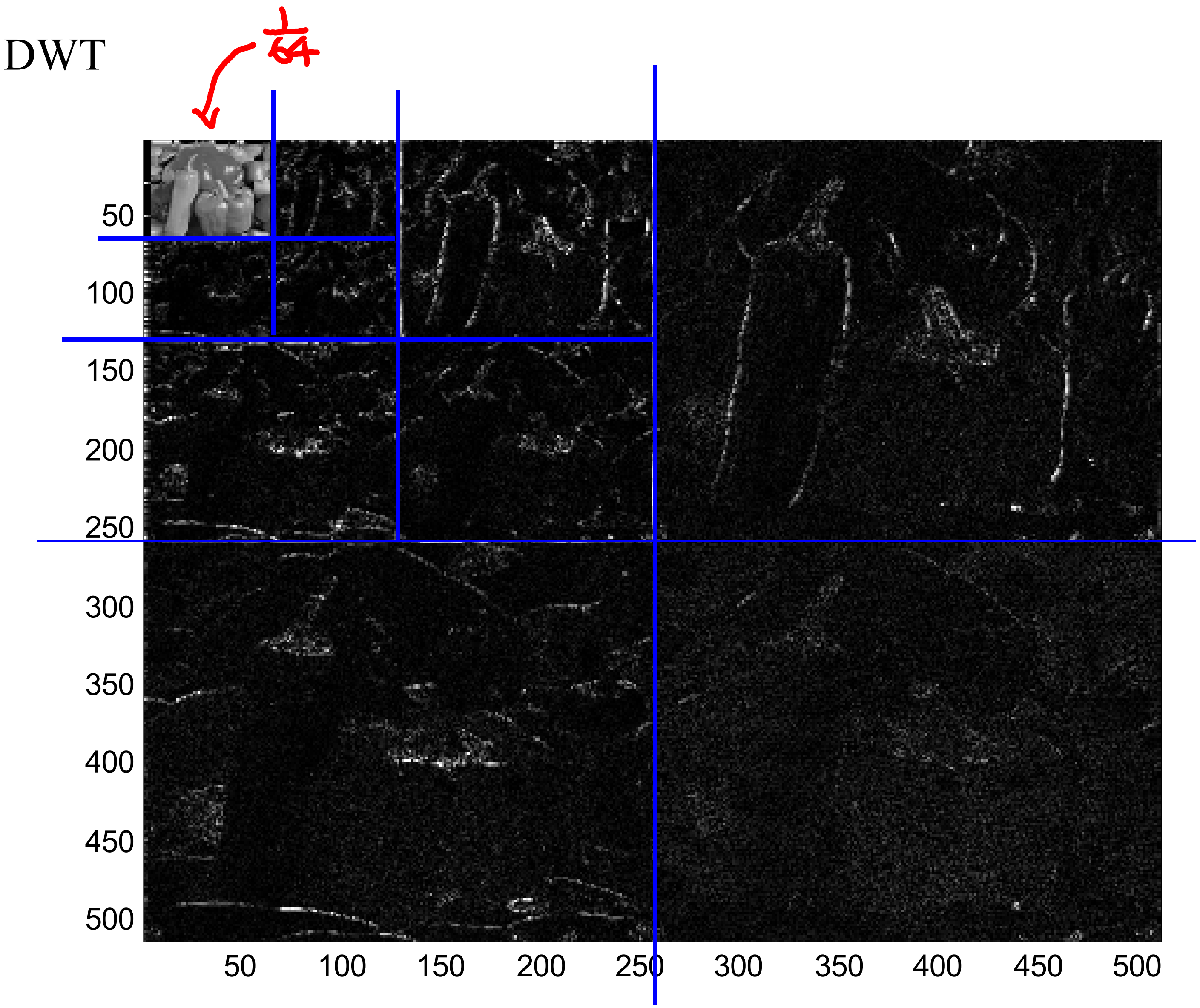


2-D DWT

的結果



3次2-D DWT 的結果



存
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)

	JPEG	JPEG2000
transform	DCT (discrete cosine transform)	wavelet
compression ratio	$\frac{1}{15}$ (gray) $\frac{1}{30}$ (color)	$\frac{1}{60}$ (gray) $\frac{1}{120}$ (color)

其他應用：edge detection

corner detection

filter design

pattern recognition

music signal processing

economical data

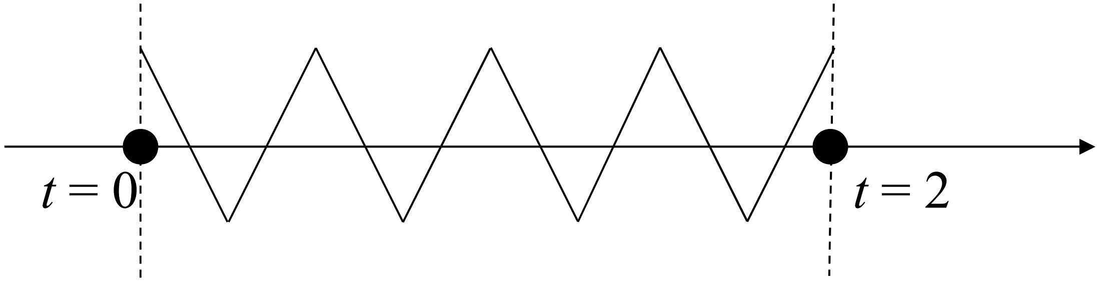
temperature analysis

feature extraction

biomedical signal processing

• Hilbert Huang Transform

Another way to determine the frequency (without the FT)



$$frequency = \frac{number\ of\ \ zero\ crossings}{2 \times \left(time\ duration \right)}$$

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
  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?

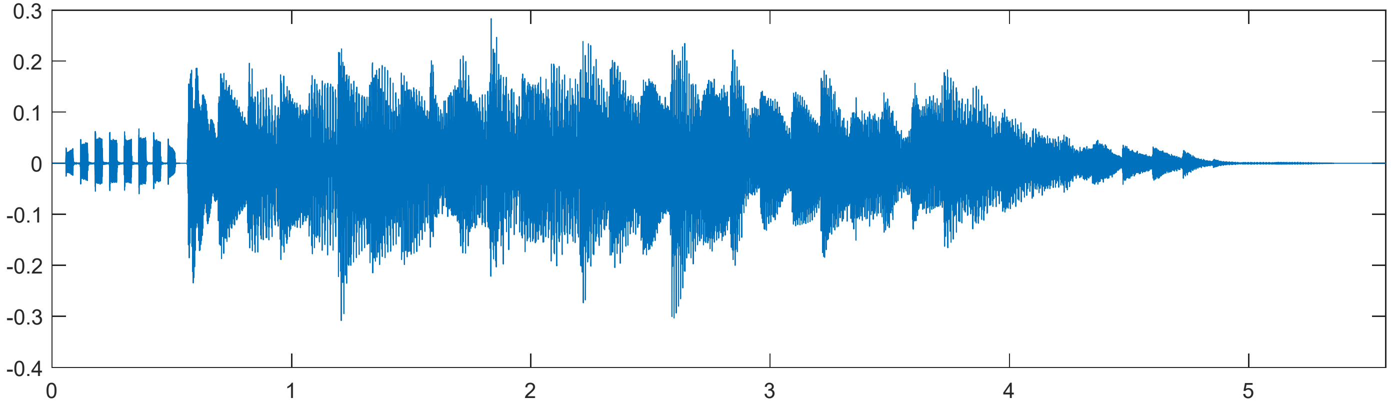
- 這個聲音檔有多少秒？

雙聲道 (**Stereo**，俗稱立體聲)

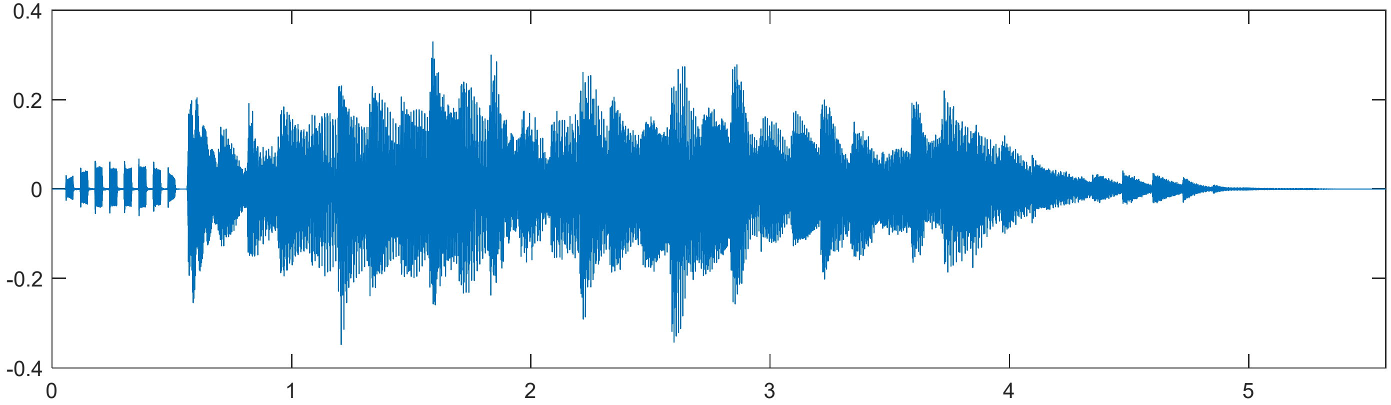
畫出聲音的波型

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)])



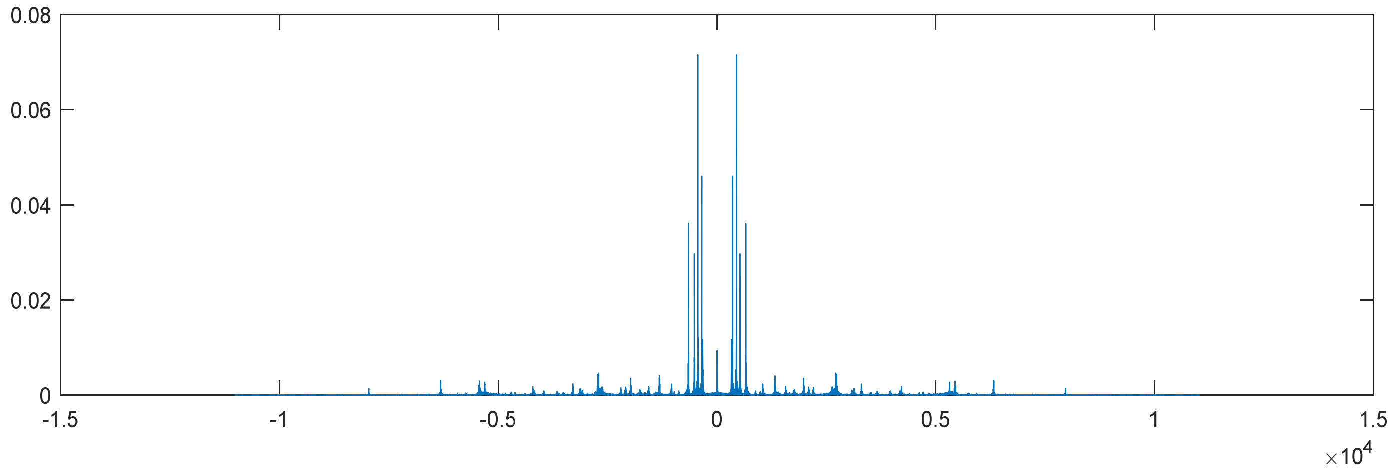
注意： *.wav 檔中所讀取的資料，值都在 –1 和 +1 之間

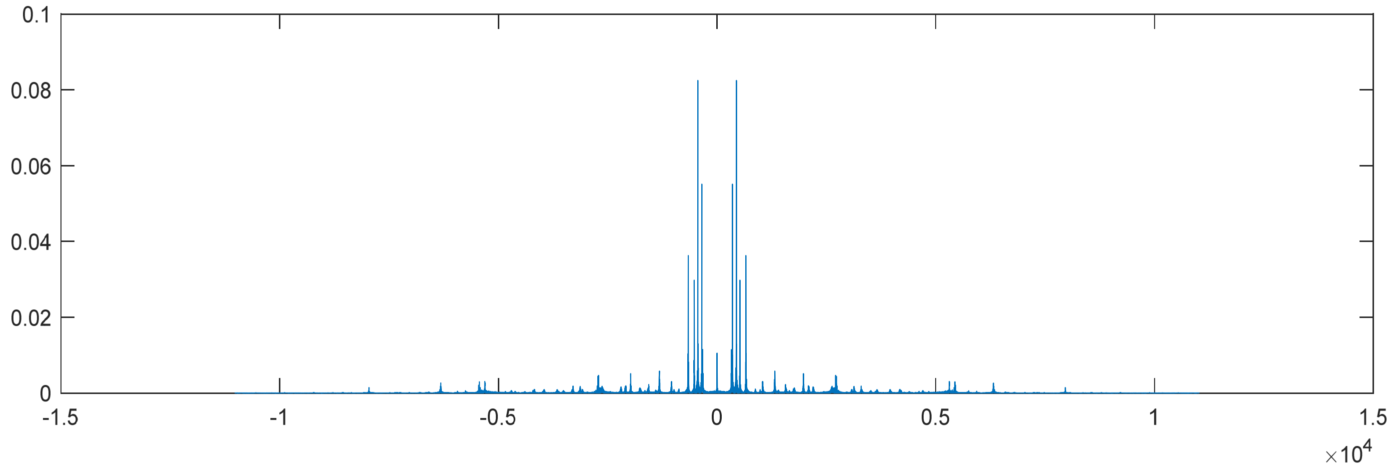
B. 繪出頻譜

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));
```

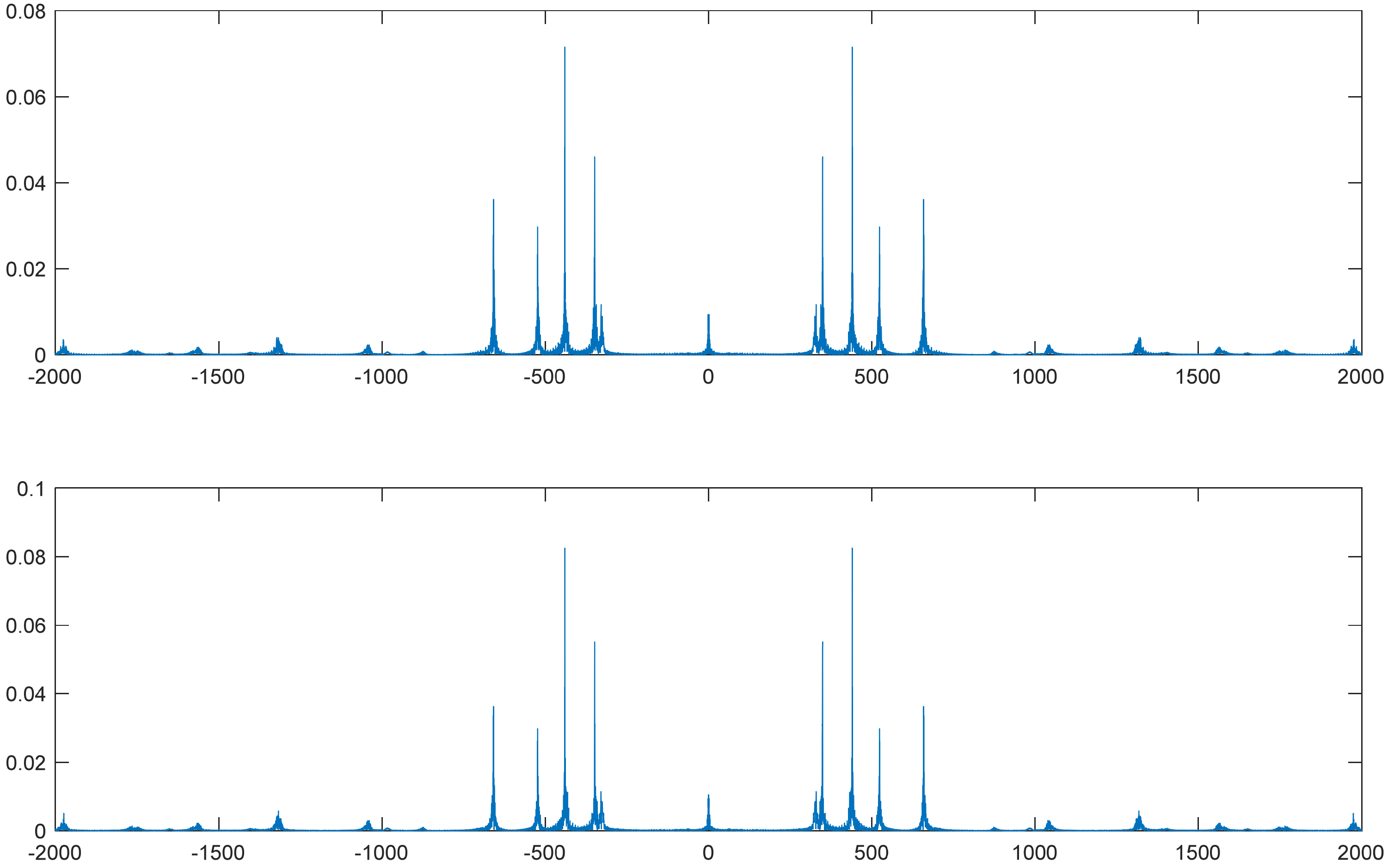
Alarm01.wav 的頻譜





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. 製作音檔： 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. 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)：

```
audioplay(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(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

附錄二 使用 Python 處理音訊的方法

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
```

後面將說明使用 Python 讀檔，畫出頻譜，撥放聲音，製作音檔，錄音的方法

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：

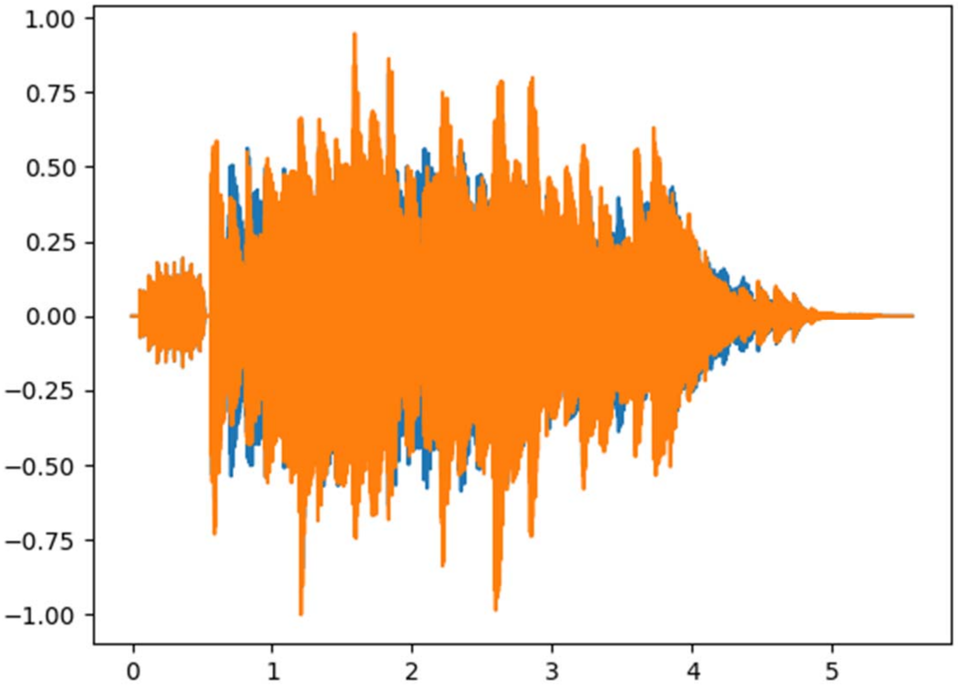
```
fs, wave_data = wavfile.read('C:/WINDOWS/Media/Alarm01.wav’)
  # fs: sampling frequency
  # If the audio file has one channel, then wave_data is a column vector
  # If the audio file has two channels, then wave_data has two column
     vectors
num_frame = len(wave_data)   # 音訊長度:
n_channel = int(wave_data.size/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

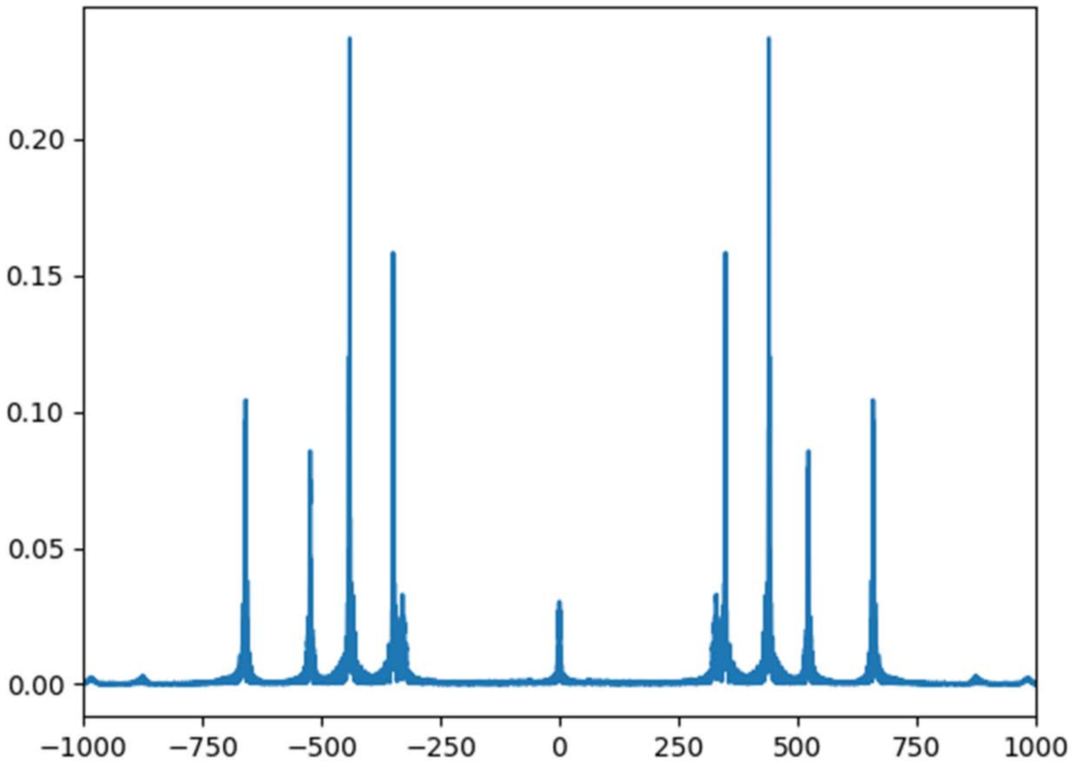
要先import 相關模組： from scipy.fftpack import fft

- 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} ~ 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/