附錄十：符合人類知覺的相似度測量工具：結構相似度 Structural Similarity (SSIM)

傳統量測兩個信號 (including images, videos, and vocal signals) 之間相似度的方式：

(1) maximal error $Max\left(\left|y[m,n]-x[m,n]\right|\right)$

(2) mean square error (MSE) $\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|y[m,n]-x[m,n]\right|^2$

power : $||^2$

energy : Σ power

(3) normalized mean square error (NMSE) $\frac{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|y[m,n]-x[m,n]\right|^2}{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|x[m,n]\right|^2}$

(4) normalized root mean square error (NRMSE) $\sqrt{\frac{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|y[m,n]-x[m,n]\right|^2}{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|x[m,n]\right|^2}}$

L_2 norm

(5) L_α-Norm

$$\|y-x\|_\alpha = \left(\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} |y[m,n]-x[m,n]|^\alpha \right)^{1/\alpha}$$

$$\frac{1}{MN}\|y-x\|_\alpha = \frac{1}{MN}\left(\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} |y[m,n]-x[m,n]|^\alpha \right)^{1/\alpha}$$

$\lim_{\alpha \to \infty} \|y-x\|_\alpha \cong \left(\text{Max}|y[m,n]-x[m,n]|^\alpha \right)^{\frac{1}{\alpha}}$

$= \text{Max}|y[m,n]-x[m,n]|$

(6) signal to noise ratio (SNR)，信號處理常用

$$10\log_{10}\left(\frac{\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} |x[m,n]|^2}{\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} |y[m,n]-x[m,n]|^2} \right)$$

dB

(7) peak signal to noise ratio (PSNR)，影像處理常用

$$10\log_{10}\left(\frac{X_{Max}^2}{\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|y[m,n]-x[m,n]\right|^2}\right)$$

X_{Max}: the maximal possible value of $x[m, n]$

In image processing, $X_{Max} = 255$

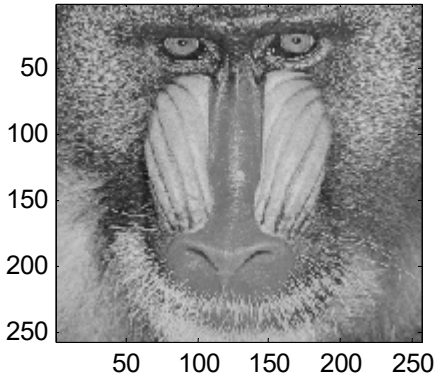
for color image:

$$10\log_{10}\left(\frac{X_{Max}^2}{\frac{1}{3MN}\sum_{R,G,B}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|y_{color}[m,n]-x_{color}[m,n]\right|^2}\right)$$

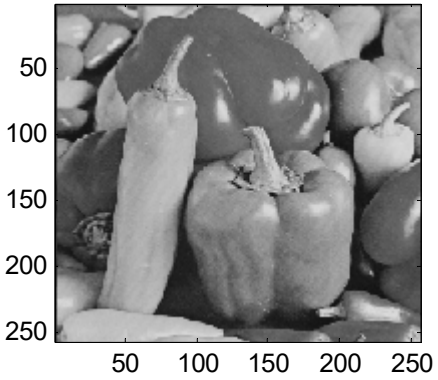
color = R, G, or B

然而，MSE 和 NRMSE 雖然在理論上是合理的，但卻無法反映出實際上兩個影像之間的相似度

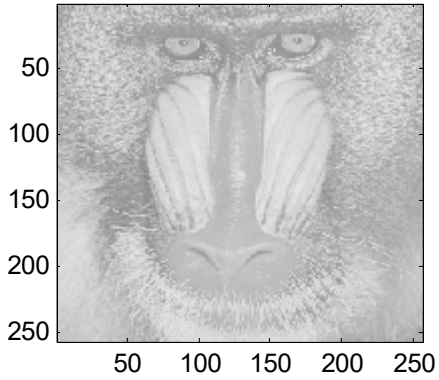
例如：以下這三張圖



圖一



圖二



圖三

圖三 = 圖一 × 0.5 + 255.5 × 0.5

照理來說，圖一和圖三較相近

然而，圖一和圖二之間的 NRMSE 為 0.4411

圖一和圖三之間的 NRMSE 為 0.4460

比較：correlation coefficient $\frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$

(8) Structural Similarity (SSIM)

有鑑於 MSE 和 PSNR 無法完全反映人類視覺上所感受的誤差，在 2004 年被提出來的新的誤差測量方法

$$SSIM(x,y) = \frac{\left(2\mu_x\mu_y + (c_1L)^2\right)}{\left(\mu_x^2 + \mu_y^2 + (c_1L)^2\right)} \frac{\left(2\sigma_{xy} + (c_2L)^2\right)}{\left(\sigma_x^2 + \sigma_y^2 + (c_2L)^2\right)}$$

$$DSSIM(x,y) = 1 - SSIM(x,y)$$

$\sigma_{xy} = E[(X-\mu_X)(Y-\mu_Y)] = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}(x[m,n]-\mu_x)(y[m,n]-\mu_y)$

$\mu_x = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}x[m,n]$

$\sigma_x^2 = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}(x[m,n]-\mu_x)^2$

μ_x, μ_y: means of x and y $\quad$ σ_x^2, σ_y^2: variances of x and y

σ_{xy}: covariance of x and y $\quad$ c_1, c_2: adjustable constants

L: the maximal possible value of x − the minimal possible value of x

$L = 255$

Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Processing*, vol. 13, no. 4, pp. 600−612, Apr. 2004.

若使用 SSIM，且前頁的 c_1, c_2 皆選為 $1/\sqrt{L}$ $= \frac{1}{\sqrt{255}}$

圖一、圖二之間的 SSIM 為 0.1040

圖一、圖三之間的 SSIM 為 0.7720

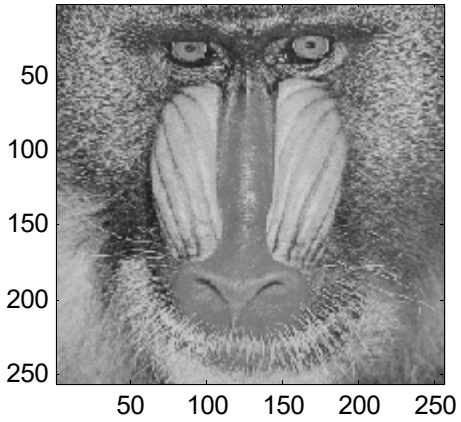
反映出了圖一、圖三之間確實有很高的相似度

比較：機率上關於相關度 (correlation) 的定義

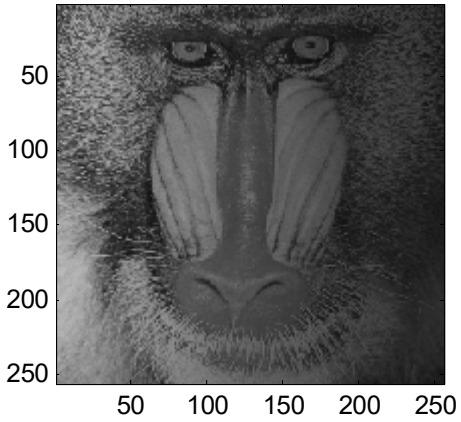
$$corr(x,y) = \frac{\sigma_{xy}}{\sigma_x \sigma_y}$$

其他幾個用 MSE 和 NRMSE 無法看出相似度，但是可以用 SSIM 看出相似度的情形

影子 shadow



圖四

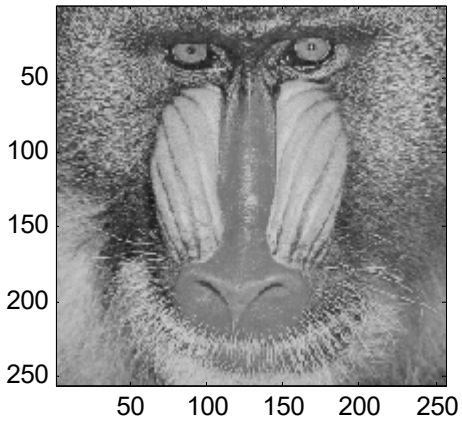


圖五

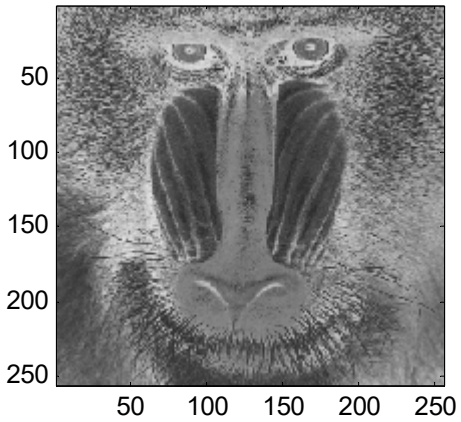
NRMSE = 0.4521 (大於圖一、圖二之間的 NRMSE)

SSIM = 0.6010

底片 the negative of a photo



圖六



圖七

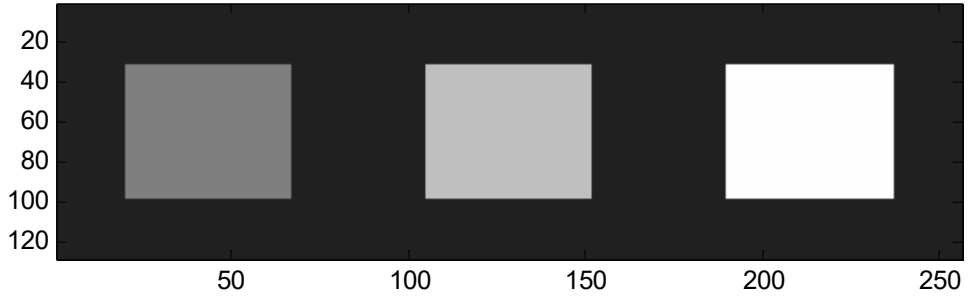
圖七 = 255 − 圖六

NRMSE = 0.5616 (大於圖一、圖二之間的 NRMSE)

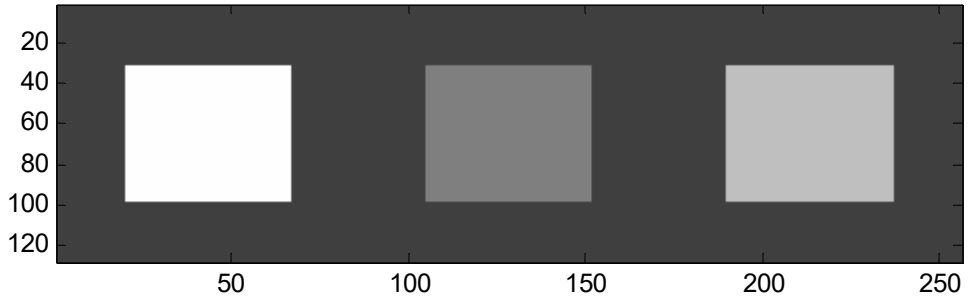
SSIM = -0.8367 (高度負相關)

同形，但亮度不同 (Same shape but different intensity)

圖八



圖九



NRMSE = 0.4978 (大於圖一、圖二之間的 NRMSE)

SSIM = 0.7333

思考：對於 vocal signal (聲音信號而言)

MSE 和 NRMSE 是否真的能反映出兩個信號的相似度？

為什麼？

metric for voice

PSEQ

IX. Basic Implementation Techniques and Fast Algorithm

◉ 9-A 快速演算法設計的原則

- **Fast Algorithm Design**

Goals: Saving Computational Time

Number of Additions

Number of Multiplications

Number of Time Cycles

Saving the Hardware Cost for Implementation

Saving the buffer size

Repeated Using a Structure

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1} & a_{M2} & \cdots & a_{MN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

M x N matrix

$y_m = a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mN}x_N$

N MULs, N-1 Adds for each m

- In sum, for an MxN matrix

MN MULs, M(N-1) ADDs are required

- $(a+jb)(c+jd) = ac - bd + j(ad+bc)$

1 complex MUL = 4 real MULs if there is no simplication

Four important concepts that should be learned from fast algorithm design:

(1) N-point DFT N can be not a power of 2

(2) Complexity of LTI Systems $y[n] = x[n] * h[n] = IFT(FT(x[n])\,FT(h[n]))$

$O(N \log N) \longrightarrow O(N)$ sectioned convolution

(3) Replacement of DFTs

(4) Simplification Techniques

ex: recursive when $h[n]$ has the form of.....

◉ 9-B 對於簡單矩陣快速演算法的設計

trivial multiplications:
乘 $\pm 2^k$ or 乘 $\pm j 2^k$

如何簡化下面四個運算

$y_1 = [a \;\; 2a]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$
2x1 matrix

(1) $y_1 = ax_1 + 2ax_2$ $\quad y_1 = a(x_1 + 2x_2)$

1 MUL, 1 ADD

(2) $\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} a & a \\ a & a \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$y_1 = a(x_1 + x_2)$
$y_2 = y_1$

1 MUL, 1 ADD

(3) $\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} a & b \\ b & a \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \frac{a+b}{2} & \frac{a+b}{2} \\ \frac{a+b}{2} & \frac{a+b}{2} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \frac{a-b}{2} & -\frac{a+b}{2} \\ -\frac{a+b}{2} & \frac{a-b}{2} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$y_1 = z_1 + z_3$
$y_2 = z_2 + z_4$

$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} \frac{a+b}{2} & \frac{a+b}{2} \\ \frac{a+b}{2} & \frac{a+b}{2} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, $\begin{bmatrix} z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} \frac{a-b}{2} & -\frac{a+b}{2} \\ -\frac{a+b}{2} & \frac{a-b}{2} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

① $z_1 = \frac{a+b}{2}(x_1 + x_2)$ 1 MUL 1 ADD
$z_2 = z_1$

② $z_3 = \frac{a-b}{2}(x_1 - x_2)$ 1 MUL 1 ADD
$z_4 = -z_3$

③ $y_1 = z_1 + z_3$
$y_2 = z_2 + z_4$ 2 ADDs

(4) $\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} a & b \\ c & a \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

3 MULs, 3 ADDs

(4)

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} a & b \\ c & a \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} a & a \\ a & a \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 & b-a \\ c-a & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} a & a \\ a & a \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$ Case(2)

$$\begin{bmatrix} z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} 0 & b-a \\ c-a & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

(i) $z_1 = a[x_1 + x_2], \quad z_2 = z_1$ 1 MUL, 1 ADD

(ii) $z_3 = (b-a)x_2, \qquad z_4 = (c-a)x_1$ 2 MULs

(iii) $y_1 = z_1 + z_3, \qquad y_2 = z_2 + z_4$ 2 ADDs

3 MULs, 3 ADDs

問題思考：如何對 complex number multiplication 來做 implementation？

$(a+jb)(c+jd) = e+jf$, $e = ac-bd$, $f = ad+bc$

$$\begin{bmatrix} e \\ f \end{bmatrix} = \begin{bmatrix} c & -d \\ d & c \end{bmatrix}\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} c & c \\ c & c \end{bmatrix}\begin{bmatrix} a \\ b \end{bmatrix} + \begin{bmatrix} 0 & -c-d \\ d-c & 0 \end{bmatrix}\begin{bmatrix} a \\ b \end{bmatrix}$$

(Case (4))

① $z_1 = c(a+b)$ 1 MUL
$z_2 = z_1$ 1 ADD

② $z_3 = (-c-d)b$ 2 MULs
$z_4 = (d-c)a$ 2 ADDs

③ $e = z_1 + z_3$
$f = z_2 + z_4$ 2 ADDs

In sum, 3 MULs, 5 ADDs are required

Specially, if $c+jd = e^{j\theta} = \cos\theta + j\sin\theta$

$$\begin{bmatrix} c & -d \\ d & c \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

if $c = \frac{\sqrt{2}}{2}$, $d = -\frac{\sqrt{2}}{2}$

$$\begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix}\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} e \\ f \end{bmatrix}$$

$e = \frac{\sqrt{2}}{2}(a+b)$
$f = \frac{\sqrt{2}}{2}(a-b)$ 2 MULs

◉ 9-C General Way for Simplifying Calculation

假設一個 $M \times N$ sub-rectangular matrix **S** 可分解為 column vector 及 row vector 相乘

$$\mathbf{S} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \end{bmatrix} \begin{bmatrix} b_1 & b_2 & \cdots & b_N \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} = S \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

$$\begin{cases} z = b_1 x_1 + b_2 x_2 + \ldots + b_N x_N \\ y_n = a_n z \end{cases}$$

若 $[a_1, a_2, \ldots, a_M]^T$ 有 M_0 個相異的 non-trivial values

$$(a_m \neq \pm 2^k, \quad a_m \neq \pm 2^k a_h \text{ where } m \neq h)$$

$[b_1, b_2, \ldots, b_N]$ 有 N_0 個相異的 non-trivial values

則 **S** 共需要 $M_0 + N_0$ 個乘法

$$\begin{bmatrix} z[1] \\ z[2] \\ \vdots \\ z[N] \end{bmatrix} = \mathbf{S} \begin{bmatrix} x[1] \\ x[2] \\ \vdots \\ x[N] \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \end{bmatrix} \begin{bmatrix} b_1 & b_2 & \cdots & b_N \end{bmatrix} \begin{bmatrix} x[1] \\ x[2] \\ \vdots \\ x[N] \end{bmatrix}$$

Step 1 $\quad z_a = b_1 x[1] + b_2 x[2] + \ldots. + b_N x[N]$

Step 2 $\quad z[1] = a_1\, z_a\,,\ z[2] = a_2\, z_a\,, \ldots\ldots\ldots, z[N] = a_M\, z_a$

簡化理論的變型

$$\mathbf{S} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \end{bmatrix} \begin{bmatrix} b_1 & b_2 & \cdots & b_N \end{bmatrix} + \mathbf{S_1}$$

$\mathbf{S_1}$ 也是一個 $M \times N$ matrix

若 $\mathbf{S_1}$ 有 P_1 個值不等於 0, 則 S 的乘法量上限為 $M_0 + N_0 + P_1$

$$\mathbf{S} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_M \end{bmatrix} \begin{bmatrix} b_1 & b_2 & \cdots & b_N \end{bmatrix} + \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_M \end{bmatrix} \begin{bmatrix} d_1 & d_2 & \cdots & d_N \end{bmatrix} + \mathbf{S_1}$$

以此類推

思考： 對於如下的情形需要多少乘法

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & f & e \\ f & e & e & f \\ d & c & b & a \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} a & d \\ d & a \end{bmatrix} \begin{bmatrix} x_1 \\ x_4 \end{bmatrix}$, $\begin{bmatrix} z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} b & c \\ c & b \end{bmatrix} \begin{bmatrix} x_2 \\ x_3 \end{bmatrix}$, $y_1 = z_1 + z_3$, $y_4 = z_2 + z_4$

Case (3), 2 MULs　　Case (3), 2 MULs

$\begin{bmatrix} y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} e & f \\ f & e \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} e & f \\ f & e \end{bmatrix} \begin{bmatrix} x_4 \\ x_3 \end{bmatrix} = \begin{bmatrix} e & f \\ f & e \end{bmatrix} \begin{bmatrix} x_1 + x_4 \\ x_2 + x_3 \end{bmatrix}$

Case (3), 2 MULs

In sum, 6 MULs are required

◉ 9-D Examples

$$\begin{bmatrix} X[0] \\ X[1] \\ \vdots \\ X[N-1] \end{bmatrix} = F \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix}, \quad m=0\sim N-1,\ n=0\sim N-1, \quad F(m,n) = e^{-j\frac{2\pi}{N}mn}$$

DFT: $$X[m] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi mn}{N}}$$

Without any simplification, the DFT needs $4N^2$ real multiplications ($x[n]$ may be complex)

- 3×3 DFT 可以用特殊方法簡化

$$\begin{matrix} & n=0 & n=1 & n=2 \\ m=0 \\ m=1 \\ m=2 \end{matrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1/2 & -1/2 \\ 1 & -1/2 & -1/2 \end{bmatrix} + j \begin{matrix} \\ m=0 \\ m=1 \\ m=2 \end{matrix} \begin{matrix} n=0 & n=1 & n=2 \\ \end{matrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & -\sqrt{3}/2 & \sqrt{3}/2 \\ 0 & \sqrt{3}/2 & -\sqrt{3}/2 \end{bmatrix}$$

$F_r: \cos\frac{2\pi}{3}mn$ — 0 MUL

$F_i: -\sin\frac{2\pi}{3}mn$ — 1 MUL

$X = F_{N\times N}\, x$, $X = X_r + jX_i$, $x = x_r + jx_i$

$(X_r + jX_i) = (F_r + jF_i)(x_r + jx_i)$

$= F_r x_r - F_i x_i + jF_i x_r + jF_r x_i$

$F_r(m,n) = \cos\frac{2\pi}{N}mn$

$F_i(m,n) = -\sin\frac{2\pi}{N}mn$

$m,n = 0,1,2,\cdots N-1$

If $\begin{bmatrix} z_0 \\ z_1 \\ z_2 \end{bmatrix} = F_i \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -a & a \\ 0 & a & -a \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix}$, $a = \frac{\sqrt{3}}{2}$

$z_0 = 0$, $z_1 = a(-x_1 + x_2)$, $z_2 = -z_1$

In sum, $2(0+1) = 2$ MULs are required.

MULs of F

= 2x MULs of Fr + 2x MULs of Fi

- 5×5 DFT 的例子

real part　　imaginary part

$$\begin{matrix} & n=0 & 1 & 2 & 3 & 4 \\ m=0 & 1 & 1 & 1 & 1 & 1 \\ m=1 & 1 & a & b & b & a \\ m=2 & 1 & b & a & a & b \\ m=3 & 1 & b & a & a & b \\ m=4 & 1 & a & b & b & a \end{matrix} - j \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & c & d & -d & -c \\ 0 & d & -c & c & -d \\ 0 & -d & c & -c & d \\ 0 & -c & -d & d & c \end{bmatrix}$$

$$a = \cos(2\pi/5)$$
$$b = \cos(4\pi/5)$$
$$c = \sin(2\pi/5)$$
$$d = \sin(4\pi/5)$$

$F_r(m,n) = \cos\left(\frac{2\pi}{5}mn\right)$　2 MULs　　$F_i(m,n) = -\sin\left(\frac{2\pi}{5}mn\right)$　3 MULs

$2(2+3) = 10$ MULs are required in sum.

$$\begin{bmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & a & b & b & a \\ 1 & b & a & a & b \\ 1 & b & a & a & b \\ 1 & a & b & b & a \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$$\begin{bmatrix} z_5 \\ z_6 \\ z_7 \\ z_8 \\ z_9 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & c & d & -d & -c \\ 0 & d & -c & c & -d \\ 0 & -d & c & -c & d \\ 0 & -c & -d & d & c \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

$z_0 = x_0 + x_1 + x_2 + x_3 + x_4$

$z_3 = z_2$, $z_4 = z_1$

$z_5 = 0$, $z_8 = -z_7$, $z_9 = -z_6$

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} x_0 \\ x_0 \end{bmatrix} + \begin{bmatrix} a & b \\ b & a \end{bmatrix} \begin{bmatrix} x_1 + x_4 \\ x_2 + x_3 \end{bmatrix}$$

2 MULs

Case (3)

$$\begin{bmatrix} z_6 \\ z_7 \end{bmatrix} = \begin{bmatrix} c & d \\ d & -c \end{bmatrix} \begin{bmatrix} x_1 - x_4 \\ x_2 - x_3 \end{bmatrix}$$

3 MULs

$$= \begin{bmatrix} d & d \\ d & d \end{bmatrix} \begin{bmatrix} x_1 - x_4 \\ -x_2 - x_3 \end{bmatrix} + \begin{bmatrix} c-d & 0 \\ 0 & -c-d \end{bmatrix} \begin{bmatrix} x_1 - x_4 \\ x_2 - x_3 \end{bmatrix}$$

8-point DCT

$$\begin{bmatrix} y[0] \\ y[1] \\ y[2] \\ y[3] \\ y[4] \\ y[5] \\ y[6] \\ y[7] \end{bmatrix} = \begin{bmatrix} 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 & 0.3536 \\ 0.4904 & 0.4157 & 0.2778 & 0.0975 & -0.0975 & -0.2778 & -0.4157 & -0.4904 \\ 0.4619 & 0.1913 & -0.1913 & -0.4619 & -0.4619 & -0.1913 & 0.1913 & 0.4619 \\ 0.4157 & -0.0975 & -0.4904 & -0.2778 & 0.2778 & 0.4904 & 0.0975 & -0.4157 \\ 0.3536 & -0.3536 & -0.3536 & 0.3536 & 0.3536 & -0.3536 & -0.3536 & 0.3536 \\ 0.2778 & -0.4904 & 0.0975 & 0.4157 & -0.4157 & -0.0975 & 0.4904 & -0.2778 \\ 0.1913 & -0.4619 & 0.4619 & -0.1913 & -0.1913 & 0.4619 & -0.4619 & 0.1913 \\ 0.0975 & -0.2778 & 0.4157 & -0.4904 & 0.4904 & -0.4157 & 0.2778 & -0.0975 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \\ x[6] \\ x[7] \end{bmatrix}$$

觀察對稱性質之後，令

$$\begin{bmatrix} z[0] \\ z[1] \\ z[2] \\ z[3] \end{bmatrix} = \begin{bmatrix} x[0]+x[7] \\ x[1]+x[6] \\ x[2]+x[5] \\ x[3]+x[4] \end{bmatrix} \qquad \begin{bmatrix} z[4] \\ z[5] \\ z[6] \\ z[7] \end{bmatrix} = \begin{bmatrix} x[0]-x[7] \\ x[1]-x[6] \\ x[2]-x[5] \\ x[3]-x[4] \end{bmatrix}$$

Part 1:
$$\begin{bmatrix} y[0] \\ y[2] \\ y[4] \\ y[6] \end{bmatrix} = \begin{bmatrix} 0.3536 & 0.3536 & 0.3536 & 0.3536 \\ 0.4619 & 0.1913 & -0.1913 & -0.4619 \\ 0.3536 & -0.3536 & -0.3536 & 0.3536 \\ 0.1913 & -0.4619 & 0.4619 & -0.1913 \end{bmatrix} \begin{bmatrix} z[0] \\ z[1] \\ z[2] \\ z[3] \end{bmatrix}$$

$$\begin{bmatrix} z[0] \\ z[1] \\ z[2] \\ z[3] \end{bmatrix} = \begin{bmatrix} x[0]+x[7] \\ x[1]+x[6] \\ x[2]+x[5] \\ x[3]+x[4] \end{bmatrix}$$

Part 2:
$$\begin{bmatrix} y[1] \\ y[3] \\ y[5] \\ y[7] \end{bmatrix} = \begin{bmatrix} 0.4904 & 0.4157 & 0.2778 & 0.0975 \\ 0.4157 & -0.0975 & -0.4904 & -0.2778 \\ 0.2778 & -0.4904 & 0.0975 & 0.4157 \\ 0.0975 & -0.2778 & 0.4157 & -0.4904 \end{bmatrix} \begin{bmatrix} z[4] \\ z[5] \\ z[6] \\ z[7] \end{bmatrix}$$

$$\begin{bmatrix} z[4] \\ z[5] \\ z[6] \\ z[7] \end{bmatrix} = \begin{bmatrix} x[0]-x[7] \\ x[1]-x[6] \\ x[2]-x[5] \\ x[3]-x[4] \end{bmatrix}$$

[Ref] B. G. Lee, "A new algorithm for computing the discrete cosine transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 32, pp. 1243-1245, Dec. 1984.

◉ 9-E Summary of the Complexity

- N-point DFT: $O(N \log_2 N)$
- N-point DCT, DST, DHT: $O(N \log_2 N)$
- Two-dimensional (2-D) $N_x \times N_y$-point DFT: $O\big((N_x N_y)\log_2(N_x N_y)\big)$ Why?
- Convolution of an M-point sequence and an N-point sequence:

$O\big((M+N-1)\log_2(M+N-1)\big)$ when M/N and N/M are not large,

$O(N)$ when $N >> M$ and M is a fixed constant.

$O(M)$ when $M >> N$ and N is a fixed constant.

- 2-D Convolution of an $(M_x \times M_y)$-point matrix and an $(N_x \times N_y)$-point matrix:

$$O\Big((M_x + N_x - 1)(M_y + N_y - 1)\log_2\big((M_x + N_x - 1)(M_y + N_y - 1)\big)\Big)$$

when M_xM_y/N_xN_y and N_xN_y/M_xM_y are not large,

$O\left(M_xM_y\right)$ when $M_xM_y >> N_xN_y$

$O\left(N_xN_y\right)$ when $N_xN_y >> M_xM_y$,
and Mx, M_y are fixed constants.

X. Fast Fourier Transform

- C. S. Burrus and T. W. Parks, “DFT / FFT and convolution algorithms”, John Wiley and Sons, New York, 1985.
- R. E. Blahut, *Fast Algorithm for Digital Signal Processing*, Addison Wesley Publishing Company.

$$X[m]=\sum_{n=0}^{N-1}x[n]e^{-j\frac{2\pi mn}{N}}$$

N-point Fourier Transform:　運算量為 N^2

FFT (with the Cooley Tukey algorithm):　運算量為 $N\log N$

要學到的概念：(1)快速演算法不是只有 Cooley Tukey algorithm

(2) 不是只有 $N=2^k$ 有時候才有快速演算法

◉ 10-A Other DFT Implementation Algorithms

(1) Cooley-Tukey algorithm (Butterfly form) $N=2^k$ $O(N \log N)$

(2) Radix-4, 8, 16, …. Algorithms

(3) Prime Factor Algorithm

(4) Goertzel Algorithm

(5) Chirp Z transform (CZT)

(6) Winograd algorithm

Reference

- J. W. Cooley and J. W. Tukey, "An algorithm for the machine computation of complex Fourier series," *Mathematics of Computation*, vol. 19, pp. 297-301, Apr. 1965. (Cooley-Tukey)
- C. S. Burrus, "Index Mappings for multidimensional formulation of the DFT and convolution," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 25, pp. 1239-242, June 1977. (Prime factor)
- G. Goertzel, "An algorithm for the evaluation of finite trigonometric series," *American Math. Monthly*, vol. 65, pp. 34-35, Jan. 1958. (Goertzel)
- C. R. Hewes, R. W. Broderson, and D. D. Buss, "Applications of CCD and switched capacitor filter technology," *Proc. IEEE*, vol. 67, no. 10, pp. 1403-1415, Oct. 1979. (CZT)
- S. Winograd, "On computing the discrete Fourier transform," *Mathematics of Computation*, vol. 32, no. 141, pp. 179-199, Jan. 1978. (Winograd)
- R. E. Blahut, *Fast Algorithm for Digital Signal Processing*, Reading, Mass., Addison-Wesley, 1985.

$$G[m]=\sum_{n=0}^{N-1} g[n]e^{-j\frac{2\pi mn}{N}}$$

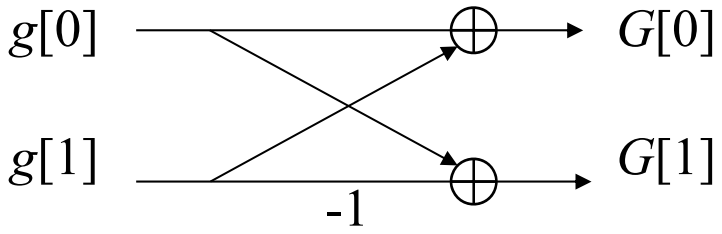
when $N=2$

$$G_1[m]=\sum_{n=0}^{1} g[n]e^{-j\pi mn}$$

$$=\sum_{n=0}^{1} g[n](-1)^{mn}$$

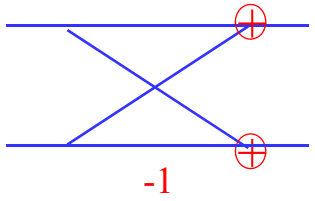
2-point DFT

$$\begin{bmatrix} G[0] \\ G[1] \end{bmatrix}=\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} g[0] \\ g[1] \end{bmatrix}$$



◉ 10-B Cooley Tukey Algorithm

When $N = 2^k$



$$X[m] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi mn}{N}}$$

$$= \sum_{n=0}^{N/2-1} x[2n] e^{-j\frac{2\pi m(2n)}{N}} + \sum_{n=0}^{N/2-1} x[2n+1] e^{-j\frac{2\pi m(2n+1)}{N}}$$

$$= \sum_{n=0}^{N/2-1} x_1[n] e^{-j\frac{2\pi mn}{N/2}} + e^{-j\frac{2\pi m}{N}} \sum_{n=0}^{N/2-1} x_2[n] e^{-j\frac{2\pi mn}{N/2}}$$

$m = 0 \sim N-1$

$\frac{N}{2}$-point DFT for $x_1[n]$

twiddle factors

$\frac{N}{2}$-point DFT for $x_2[n]$

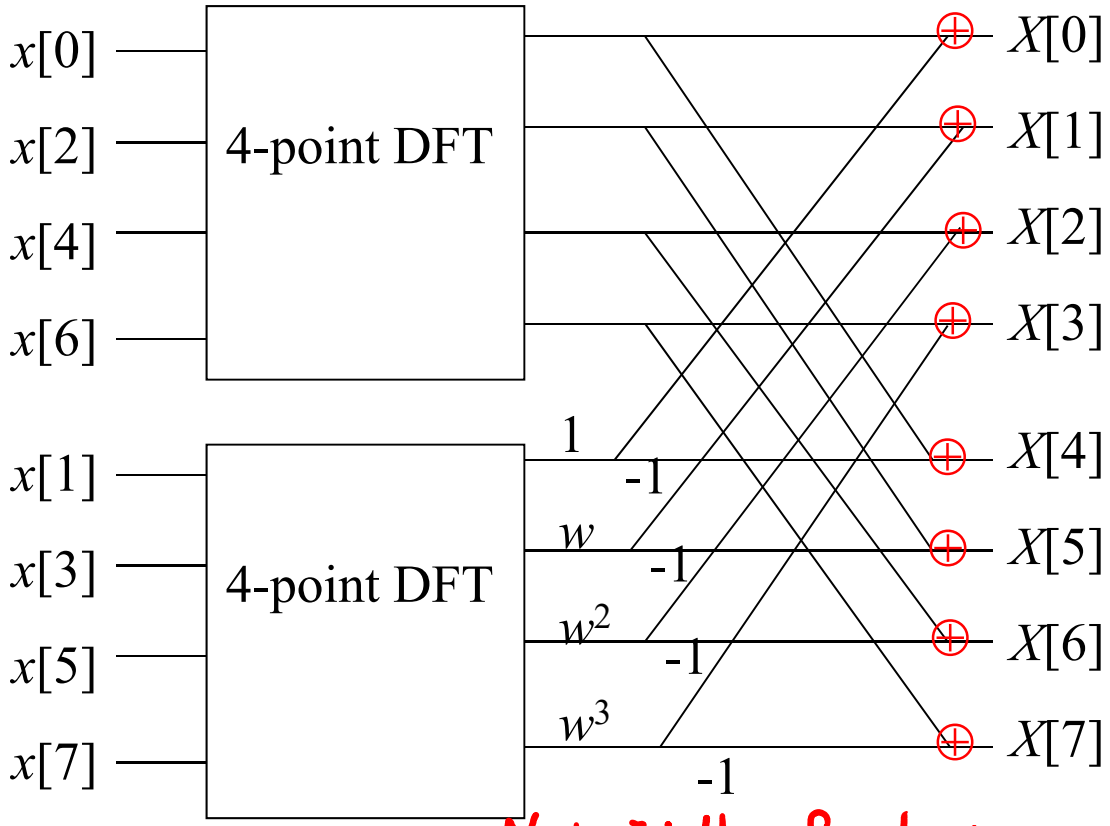
$$x_1[n] = x[2n], \quad x_2[n] = x[2n+1]$$

Therefore,

one N-point DFT = two $(N/2)$-point DFTs + twiddle factors

$\frac{N}{2}$ + twiddle factors

8-point DFT



$\frac{N}{2}$ twiddle factors

$m: 0 \sim N-1$

$w = e^{-j\frac{2\pi}{N}}$ twiddle factor: $e^{-j\frac{2\pi m}{N}} = w^m$

$$w^{(m+\frac{N}{2})} = e^{-j\frac{2\pi}{N}(m+\frac{N}{2})} = e^{-j\frac{2\pi}{N}m} e^{-j\frac{2\pi}{N}\frac{N}{2}} = -e^{-j\frac{2\pi}{N}m} = -w^m$$

$e^{-j\pi} = -1$

if $N=8$

$w^{m+4} = -w^m$

When $N = 8$

$w = e^{-j\frac{2\pi}{8}}$

$w^8 = 1$

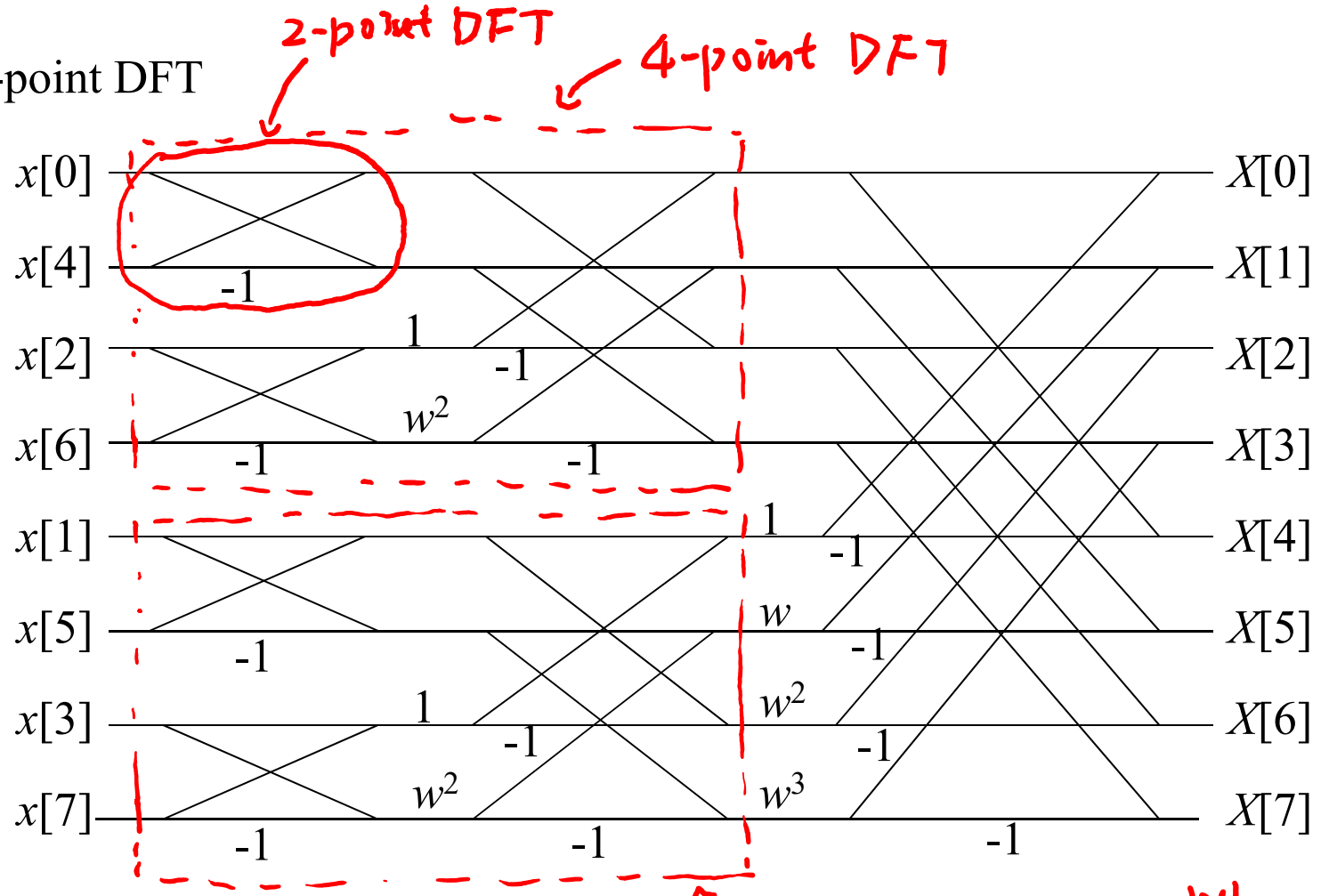
$w^4 = -1$

$w^5 = -w$

$w^6 = -w^2$

$w^7 = -w^3$

8-point DFT



$$w = e^{-j\frac{2\pi}{8}}$$

$w^2 = e^{-j\frac{\pi}{2}} = -j$

$w = e^{-j\frac{\pi}{4}} = \frac{\sqrt{2}}{2} - j\frac{\sqrt{2}}{2}$

$w^3 = \frac{\sqrt{2}}{2} + j\frac{\sqrt{2}}{2}$

When $N=4$

twiddle factor

$= e^{-j\frac{2\pi}{4}m}$

$= \left(e^{-j\frac{2\pi}{8}}\right)^{2m}$

$= w^{2m}$

- Number of real multiplications 的估算

2^k-point DFT 一共有 k 個 stages　　k-1 次 decomposition

每個 stage 和下一個 stage 之間有 2^{k-1} 個 twiddle factors

所以，一共有 $2^{k-1}(k-1)$ 個 twiddle factors　　$3 \cdot 2^{k-1}(k-1)$ real MULs

$k=3$, $3 \times 4 \times 2 = 24$

一般而言，每個 twiddle factor 需要 3 個 real multiplications

∴ 2^k-point DFT 需要

$k = \log_2 N$

$2^{k-1}(k-1) = \frac{N}{2}(\log_2 N - 1) \cong \frac{N}{2}\log_2 N$

$$3\left(2^{k-1}(k-1)\right) = \frac{3}{2}N\left(\log_2 N - 1\right)$$ 個 real multiplications

Complexity of the N-point DFT: $O(N\log_2 N)$

- 8-point DFT 只需要 4 個 real multiplications (Why?)

- 更精確的分析，使用 Cooley-Tukey algorithm 時，N-point DFT 需要

$$\frac{3}{2}N\log_2 N - 5N + 8$$ 個 real multiplications

(Why?)

◉ 10-C Radix-4 Algorithm

限制： $N = 4^k$

or $N = 2 \cdot 4^k$ (此時 Cooley-Tukey algorithm 和 radix-4 algorithm 並用)

4-point DFT

$$X[m] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi mn}{N}}$$

$$= \sum_{n=0}^{N/4-1} x[4n] e^{-j\frac{2\pi mn}{N/4}} + e^{-j\frac{2\pi m}{N}} \sum_{n=0}^{N/4-1} x[4n+1] e^{-j\frac{2\pi mn}{N/4}}$$

$$+ e^{-j\frac{2\pi(2m)}{N}} \sum_{n=0}^{N/4-1} x[4n+2] e^{-j\frac{2\pi mn}{N/4}} + e^{-j\frac{2\pi(3m)}{N}} \sum_{n=0}^{N/4-1} x[4n+3] e^{-j\frac{2\pi mn}{N/4}}$$

twiddle factors

$$e^{-j\frac{2\pi}{4}mn} = e^{-j\frac{\pi}{2}mn} = (-j)^{mn}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

One N-point DFT = four $(N/4)$-point DFTs + twiddle factors

$\frac{3}{4}N$ 個 twiddle factors

Note:

(1) radix-4 algorithm 最後可將 $N = 4^k$-point DFT 拆解成 4-point DFTs 的組合

4-point DFTs 不需要任何的乘法

(2) 使用 radix-4 algorithm 時，N-point DFT 需要

$$\frac{9}{4} N \log_4 N - \frac{43}{12} N + \frac{16}{3}$$ 個 real multiplications

- Number of real multiplications for the N-point DFT

$MUL_{20} = 4MUL_5 + 5MUL_4 = 4\times10+5\times0=40$
$MUL_{15} = 3MUL_5 + 5MUL_3 = 3\times10+5\times2=40$
$MUL_{21} = 3MUL_7 + 7MUL_3 = 3\times16+7\times2=62$
$MUL_{25} = 5MUL_5 + 5MUL_5 + 3\times(5-1)(5-1)$
$= 5\times10+5\times10+48=148$
$MUL_{49} = 7MUL_7 + 7MUL_7 + 3\times6\times6$
$= 7\times16\times2+108=332$

N	乘法數	加法數	N	乘法數	N	乘法數	N	乘法數
1	0	0	11	40	24	28	39	182
2	0	4	12	8	25	148	40	100
3	2	12	13	52	26	104	42	124
4	0	16	14	32	27	114	44	160
5	10	34	15	40	28	64	45	170
6	4	36	16	20	30	80	48	92
7	16	72	18	32	32	72	52	208
8	4	52	20	40	33	142	54	228
9	16	72	21	62	35	150	56	156
10	20	88	22	80	36	64	60	160

49 332

$90 = 2\times5\times9 = 10\times9$

$MUL_{90} = 10MUL_9 + 9MUL_{10}$

$= 10\times16 + 9\times20$

$= 340$

$65 = 5\times13$

$MUL_{65} = 5MUL_{13} + 13MUL_5$

$= 5\times52 + 13\times10$

$= 390$

$MUL_{231} = ?$

$MUL_{400} = ?$

N	乘法數	N	乘法數	N	乘法數	N	乘法數
63	256	96	280	192	752	360	1540
64	204	104	468	204	976	420	2080
65	390						
66	284	108	456	216	1020	480	2360
70	300	112	396	224	1016	504	2300
72	164	120	380	240	940	512	3180
80	260	128	560	252	1024	560	3100
81	480	144	436	256	1308	672	3496
84	248	160	680	288	1160	720	3620
88	364	168	580	312	1608	784	4412
90	340	180	680	336	1412	840	4580

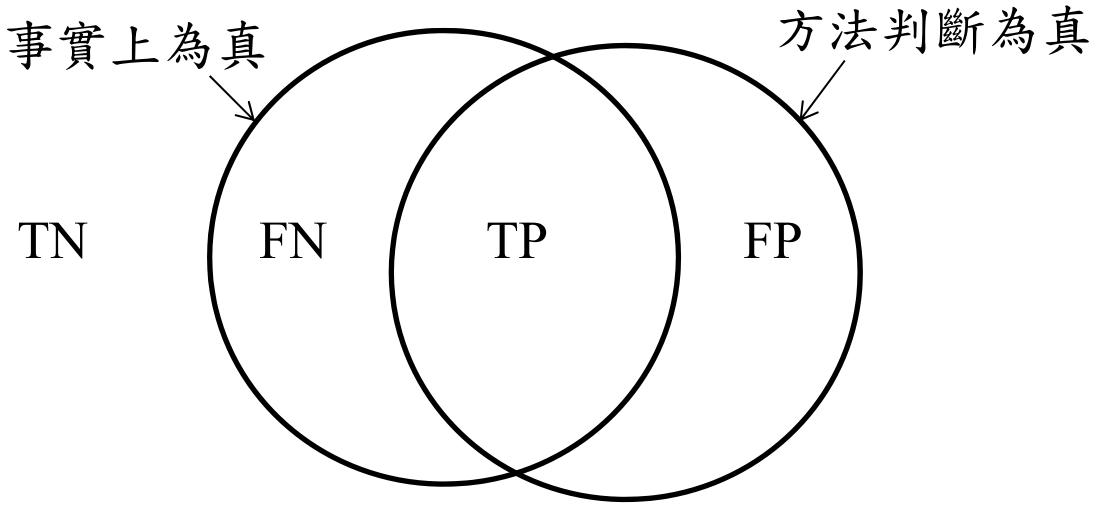
$144 = 2^4\times3^2$

N	乘法數	*N*	乘法數	*N*	乘法數	*N*	乘法數
1008	5356	1440	8680	2520	16540	4032	29488
1024	7436	1680	10420	2688	19108	4096	37516
1152	7088	2016	12728	2880	20060	4368	35828
1260	7640	2048	16836	3369	24200	4608	36812
1344	8252	2304	15868	3920	29900	5040	36860

$5040 = 7!$

$= 2^4 \times 3^2 \times 5 \times 7$

附錄十一：量測方法的精確度常用的指標



TP (true positive): 事實上為真，而且被我們的方法判斷為真的情形

FN (false negative): 事實上為真，卻未我們的方法被判斷為真的情形

FP (false positive): 事實上不為真，卻被我們的方法誤判為真的情形

TN (true negative): 事實上不為真，而且被我們的方法判斷成不為真的情形

$$precision = \frac{TP}{TP + FP} = +P \quad \text{(positive prediction rate)}$$

$$recall = \frac{TP}{TP + FN}$$

$$specificity = \frac{TN}{TN + FP} \qquad sensitivity = \frac{TP}{TP + FN} = recall$$

以抓犯人為例，TP 是有罪而且被抓到的情形，FP是無罪但被誤抓的情形，FN 是有罪但沒被抓到的情形，TN 是無罪且未被誤逮的情形

寧可錯抓一百，也不可放過一個

$\longrightarrow$ recall 高，但 precision 低

寧可錯放一百，也不可冤枉一個

$\longrightarrow$ precision 高，但 recall 低

Accuracy $$\frac{TP + TN}{TP + FP + TN + FN}$$

Detection error rate $$\frac{FP + FN}{TP + FN}$$

F-score $$2\frac{precision \times recall}{precision + recall}$$

General form of the F-score $$\frac{(1 + \beta^2)\, precision \times recall}{\beta^2\, precision + recall}$$

◉ 10-D Prime Factor Algorithm

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

N 可以是任意整數

If $N = P_1^{k_1} P_2^{k_2} \cdots\cdots P_M^{k_M}$

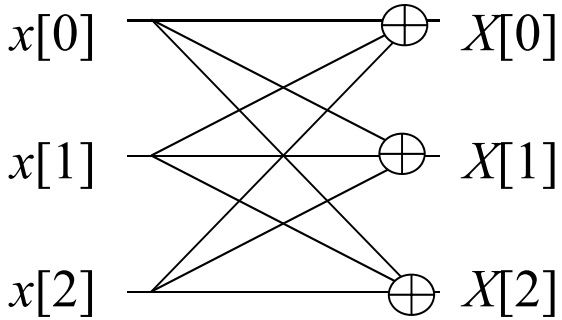
$P_1, P_2, P_3 \ldots\ldots P_M$ 不一定是 prime number, 但彼此互質

$P_1, P_2, \ldots, P_M$ are small integers and prime to each other

the powers $k_1, k_2, \ldots, k_M$ are small

then using the prime factor FFT to implement the N-point DFT may require fewer real multiplications.

3-point DFT butterfly:



Needs 4 complex multiplications (12 real multiplications)

N–point DFT butterfly: needs 3(*N*–1)(*N*–1) real multiplications

然而，可以使用特殊的方法，讓 *N*–point DFT 的乘法量大幅減少
(即使 $N \neq 2^k$)

例如 pages 346, 347, 353, 354

- Detail of the implementation method of the prime factor algorithm

$$F[m]=\sum_{n=0}^{N-1} f[n]e^{-j\frac{2\pi}{N}mn} \qquad n=0,\ 1,\ \ldots.,\ N-1,\quad m=0,\ 1,\ \ldots.,\ N-1$$

If P_1 is prime to P_2 we can find two integers a, b such that

$aP_1+bP_2=1$

$akP_1+bkP_2=k$

Case 1: Suppose that $N=P_1\times P_2$, P_1 is prime to P_2

⇩

拆成 P_2 個 P_1-point DFTs，和 P_1 個 P_2-point DFTs

當 P_1, P_2 互質時，必可找到 n_1, n_2 使得

m_1 m_2

$n=((n_1P_1+n_2P_2))_N$ $m=((m_1P_1+m_2P_2))_N$

$((\,))_N$：除以 N 的餘數

$n_1, m_1=0,\ 1,\ \ldots.,\ P_2-1,$ $n_2, m_2=0,\ 1,\ \ldots.,\ P_1-1$

且每一個 n_1, n_2 對應到唯一一個 n

$0\le n_1\le 6$, $0\le n_2\le 2$

$10=((n_1\cdot 3+n_2\cdot 7))_{21}$

$n_1=?$ $n_2=?$

(Proof):

First, if P_1 is prime to P_2, we can always find two integers d_1 and d_2 such that

$$d_1 P_1 + d_2 P_2 = 1$$

Therefore, for any n,

$$d_1 n P_1 + d_2 n P_2 = n$$

Suppose that

$$\left(\left(d_1 n\right)\right)_{P_2} = n_1, \quad \left(\left(d_2 n\right)\right)_{P_1} = n_2$$

then

$$d_1 n = n_1 + k_1 P_2, \quad d_2 n = n_2 + k_2 P_1$$

$$d_1 n P_1 + d_2 n P_2 = n_1 P_1 + n_2 P_2 + \left(k_1 + k_2\right) P_1 P_2 = n$$

$$n_1 P_1 + n_2 P_2 = n - \left(k_1 + k_2\right) N$$

If $0 \le n \le N-1$, then

$$\left(\left(n_1 P_1 + n_2 P_2\right)\right)_N = n$$

例子：當 $N=15, P_1=3, P_2=5,$

$0=((0\cdot P_1+0\cdot P_2))_{15}$

$1=((2\cdot P_1+2\cdot P_2))_{15}$

$2=((4\cdot P_1+1\cdot P_2))_{15}$

$3=((1\cdot P_1+0\cdot P_2))_{15}$

$4=((3\cdot P_1+2\cdot P_2))_{15}$

$5=((0\cdot P_1+1\cdot P_2))_{15}$

$6=((2\cdot P_1+0\cdot P_2))_{15}$

$7=((4\cdot P_1+2\cdot P_2))_{15}$

$8=((1\cdot P_1+1\cdot P_2))_{15}$

$9=((3\cdot P_1+0\cdot P_2))_{15}$

$10=((0\cdot P_1+2\cdot P_2))_{15}$

$11=((2\cdot P_1+1\cdot P_2))_{15}$

$12=((4\cdot P_1+0\cdot P_2))_{15}$

$13=((1\cdot P_1+2\cdot P_2))_{15}$

$14=((3\cdot P_1+1\cdot P_2))_{15}$

$((16P_1-15P_1+(-8)P_2+9P_2))_{15}=8$

$((1\cdot P_1+1\cdot P_2))_{15}=1$

$2P_1+(-1)P_2=1$

$16P_1+(-8)P_2=8$

$((2P_1+3P_2+(-1)P_2))_{15}=1$

$=((2P_1+2P_2))_{15}=1$ $(3P_2=15)$

$$F[m]=\sum_{n=0}^{N-1} f[n]e^{-j\frac{2\pi}{N}mn} \qquad N=P_1\times P_2$$

$$m=((m_1P_1+m_2P_2))_N=m_1P_1+m_2P_2+c_1N$$

$$n=((n_1P_1+n_2P_2))_N=n_1P_1+n_2P_2+c_2N$$

$$\begin{aligned}
e^{-j\frac{2\pi}{N}mn} &= e^{-j\frac{2\pi}{N}(m_1P_1+m_2P_2+c_1N)(n_1P_1+n_2P_2+c_2N)} \\
&= e^{-j\frac{2\pi}{N}[(m_1P_1+m_2P_2)(n_1P_1+n_2P_2)]}e^{-j\frac{2\pi}{N}[c_1N(n_1P_1+n_2P_2)]}e^{-j\frac{2\pi}{N}[c_2N(m_1P_1+m_2P_2)]}e^{-j\frac{2\pi}{N}[c_1c_2N^2]} \\
&= e^{-j\frac{2\pi}{N}[(m_1P_1+m_2P_2)(n_1P_1+n_2P_2)]}
\end{aligned}$$

$$F\left[((m_1P_1+m_2P_2))_N\right]=\sum_{n=0}^{N-1} f[((n_1P_1+n_2P_2))_N]\,e^{-j\frac{2\pi}{P_1P_2}(m_1P_1+m_2P_2)(n_1P_1+n_2P_2)}$$

$$=\sum_{n=0}^{N-1} f[((n_1P_1+n_2P_2))_N]\,e^{-j\frac{2\pi}{P_1P_2}(m_1n_1P_1P_1+m_2n_2P_2P_2+m_1n_2P_1P_2+m_2n_1P_2P_1)}$$

$$=\sum_{n=0}^{N-1} f[((n_1P_1+n_2P_2))_N]\,e^{-j\frac{2\pi}{P_2}m_1P_1n_1}\,e^{-j\frac{2\pi}{P_1}m_2P_2n_2}$$

$$=\sum_{n_2=0}^{P_1-1}\left\{\sum_{n_1=0}^{P_2-1} f[((n_1P_1+n_2P_2))_N]\,e^{-j\frac{2\pi}{P_2}m_1P_1n_1}\right\}e^{-j\frac{2\pi}{P_1}m_2P_2n_2}$$

Step 2

Step 3

$N: P_1P_2$, P_1 is prime to P_2

P_2-point DFT

P_1-point DFT

$n_1, m_1 = 0, 1, \ldots., P_2-1, \quad n_2, m_2 = 0, 1, \ldots., P_1-1$

Step 1 令 $g[n_1,n_2]=f[((n_1P_1+n_2P_2))_N]$

Step 2 固定 n_2，對 n_1 做 P_2-point DFT

$$\hat{G}_1[m_3,n_2]=\sum_{n_1=0}^{P_2-1}g[n_1,n_2]e^{-j\frac{2\pi}{P_2}m_3n_1}$$

$$G_1[m_1,n_2]=\hat{G}_1[((P_1m_1))_{P_2},n_2]$$

n_2 有 P_1 個值，所以有 P_1 個 P_2-point DFTs

Step 3 固定 m_3，對 n_2 做 P_1-point DFT

$$\hat{G}_2[m_1,m_4]=\sum_{n_2=0}^{P_1-1}G_1[m_1,n_2]e^{-j\frac{2\pi}{P_1}m_4n_2}$$

$$G_2[m_1,m_2]=\hat{G}_2[m_1,((P_2m_2))_{P_1}]$$

m_3 有 P_2 個值，所以有 P_2 個 P_1-point DFTs

假設 p-point DFT 要 MUL_p 個乘法

$MUL_N: P_1 MUL_{P_2}+P_2 MUL_{P_1}$

Step 4 $F[((m_1P_1+m_2P_2))_N]=G_2[m_1, m_2]$

$$F[m]=\sum_{n=0}^{N-1} f[n]e^{-j\frac{2\pi}{N}mn} \qquad n=0,\,1,\,\ldots.,\,N-1,\quad m=0,\,1,\,\ldots.,\,N-1$$

Case 2: Suppose that $N=P_1\times P_2$ ， P_1 is not prime to P_2

⇩

拆成 P_2 個 P_1-point DFTs， P_1 個 P_2-point DFTs ，和 twiddle factors

令 $n=n_1P_1+n_2$ $\qquad m=m_1+m_2P_2$

$$n_1,\,m_1=0,\,1,\,\ldots.,\,P_2-1,\quad n_2,\,m_2=0,\,1,\,\ldots.,\,P_1-1$$

$$F[m_1+m_2P_2]=\sum_{n=0}^{N-1} f[n_1P_1+n_2]e^{-j\frac{2\pi}{N}(m_1+m_2P_2)(n_1P_1+n_2)}$$

$$=\sum_{n=0}^{N-1} f[n_1P_1+n_2]e^{-j\frac{2\pi}{P_1P_2}(m_1n_1P_1+m_1n_2+m_2n_1P_1P_2+m_2n_2P_2)}$$

P_2-point DFT　　P_1-point DFT

$$=\sum_{n=0}^{N-1} f[n_1P_1+n_2]e^{-j\frac{2\pi}{P_2}m_1n_1}e^{-j\frac{2\pi}{P_1}m_2n_2}e^{-j\frac{2\pi}{P_1P_2}m_1n_2}$$

$$=\sum_{n_2=0}^{P_1-1}\left\{\sum_{n_1=0}^{P_2-1} f[n_1P_1+n_2]e^{-j\frac{2\pi}{P_2}m_1n_1}\right\}e^{-j\frac{2\pi}{N}m_1n_2}e^{-j\frac{2\pi}{P_1}m_2n_2}$$

$$F[m_1 + m_2 P_2] = \sum_{n_2=0}^{P_1-1} \left\{ \sum_{n_1=0}^{P_2-1} f[n_1 P_1 + n_2] e^{-j\frac{2\pi}{P_2} m_1 n_1} \right\} e^{-j\frac{2\pi}{N} m_1 n_2} e^{-j\frac{2\pi}{P_1} m_2 n_2}$$

$n_2 = 0, 1, \cdots P_1 - 1$ P_1 個 P_2-point DFTs → Step 2

Step 3

Step 4 ← $m_1 = 0, 1, \cdots, P_2 - 1$ P_2 個 P_1-point DFTs

$e^{-j\frac{2\pi}{N} m_1 n_2}$ 被稱為 twiddle factor，需要額外的乘法

m_1 有 P_2 個

n_2 有 P_1 個

$n_1, m_1 = 0, 1, \ldots, P_2 - 1,$ $n_2, m_2 = 0, 1, \ldots, P_1 - 1$

Number of twiddle factors: $P_1 \times P_2 = N$

excluding the case where $m_1 = 0$ or $n_2 = 0$

Number of twiddle factors: $(P_1 - 1) \times (P_2 - 1)$

Step 1 令 $g[n_1,n_2]=f[n_1P_1+n_2]$

Step 2 固定 n_2，對 n_1 作 P_2-point DFT

$$G_1[m_1,n_2]=\sum_{n_1=0}^{P_2-1} g[n_1,n_2]e^{-j\frac{2\pi}{P_2}m_1n_1}$$

n_2 有 P_1 個值，所以有 P_1 個 P_2-point DFTs

Step 3 $G_2[m_1,n_2]=G_1[m_1,n_2]e^{-j\frac{2\pi}{N}m_1n_2}$ twiddle factors

Step 4 固定 m_1，對 n_2 做 P_2 個 P_1-point DFT

$$G_3[m_1,m_2]=\sum_{n_1=0}^{P_1-1} G_2[m_1,n_2]e^{-j\frac{2\pi}{P_1}m_2n_2}$$

m_1 有 P_2 個值，所以有 P_2 個 P_1-point DFTs

Step 5 $F[m_1+m_2P_2]=G_3[m_1,m_2]$

$MUL_N = P_1 MUL_{P_2} + P_2 MUL_{P_1} + 3(P_1-1)(P_2-1)$

If N is not a multiple of 2 or 3

◉ 10-E FFT 的乘法量的計算

假設 $N = P_1 \times P_2$ ， P_1 is **prime** to P_2

P_1-point DFT 的乘法量為 B_1， P_2-point DFT 的乘法量為 B_2

則 N-point DFT 的乘法量為

$$P_2B_1 + P_1B_2$$

假設 $N = P_1 \times P_2 \times \cdots\cdots \times P_K$ $P_1, P_2, \ldots, P_K$ 彼此互質

P_k-point DFT 的乘法量為 B_k

則 N-point DFT 可分解成 (N/P_1) 個 P_1-point DFTs

(N/P_2) 個 P_2-point DFTs

:

(N/P_K) 個 P_K-point DFTs

總乘法量為

$$\frac{N}{P_1}B_1 + \frac{N}{P_2}B_2 + \cdots\cdots + \frac{N}{P_k}B_k$$

假設 $N = P_1 \times P_2$ ， P_1 is **not prime** to P_2

P_1-point DFT 的乘法量為 B_1， P_2-point DFT 的乘法量為 B_2

則 N-point DFT 的乘法量為

twiddles: $e^{-j\frac{2\pi}{N}m_1 n_2}$

且 $m_1 n_2$ 當中 $(m_1 = 0, 1, \ldots, P_2 - 1, \quad n_2 = 0, 1, \ldots, P_1 - 1)$

有 D_1 個值不為 $N/12$ 及 $N/8$ 的倍數

有 D_2 個值為 $N/12$ 或 $N/8$ 的倍數，但不為 $N/4$ 的倍數

則 N-point DFT 的乘法量為

$$\boldsymbol{P_2 B_1 + P_1 B_2 + 3D_1 + 2D_2}$$

Note: $a \times \exp(j\,\theta)$，當 a 為 complex，需要 3 個乘法

然而，當 $\theta = \pi/4$，只需 2個乘法

當 $\theta = \pi/3$，只需 2 個乘法

例子： 16-point DFT, $16 = 8 \times 2$,

$$\text{乘法量} = 2 \times 4 + 8 \times 0 + 3 \times\ 4 + 2 \times\ 2 = 24$$

$$16 = 4 \times 4$$

$$\text{乘法量} = 4 \times 0 + 4 \times 0 + 3 \times\ 4 + 2 \times\ 4 = 20$$

◉ 10-F Goertzel Algorithm

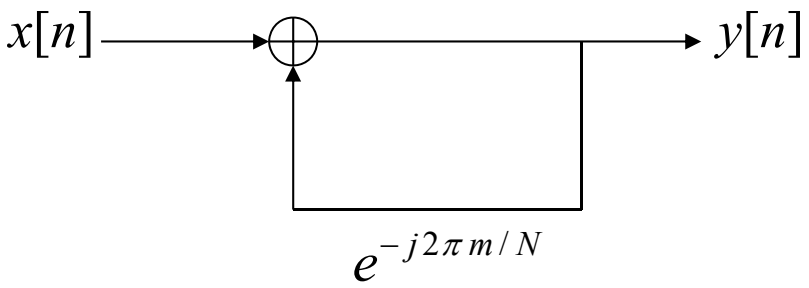
DFT: $F[m]=\sum_{n=0}^{N-1} f[n]e^{-j\frac{2\pi}{N}mn}$

$x[n]=f[N-n],\ n=1, 2, \ldots, N$

$$F[m]=x[1]e^{-j\frac{2\pi}{N}m(N-1)}+x[2]e^{-j\frac{2\pi}{N}m(N-2)}+\cdots+x[N]e^{-j\frac{2\pi}{N}m(0)}$$

$f[N-1]$ $f[N-2]$ $f[0]$

$F[m]=y[N]$



優點： Hardware 最為精簡

缺點： 運算時間較長 (N-1 times of feedback)

◉ 10-G Chirp Z Transform

當$\Delta_t \Delta_f = 1/N$時， Continuous Fourier transform可以用DFT和FFT來做implementation。

問題: 當$\Delta_t \Delta_f \neq 1/N$ 時怎麼辦?

$$G(f) = \int e^{-j2\pi f t} g(t)\,dt \xrightarrow[f = m\,\Delta_f]{\text{取 } t = n\,\Delta_t} G(m\Delta_f) = \Delta_t \sum_n e^{-j2\pi mn\Delta_t\Delta_f} g(n\Delta_t)$$

$$G(m\Delta_f) = \Delta_t e^{-j\pi m^2\Delta_t\Delta_f} \sum_n e^{j\pi(m-n)^2\Delta_t\Delta_f} e^{-j\pi n^2\Delta_t\Delta_f} g(n\Delta_t)$$

$$G(m\Delta_f) = \Delta_t e^{-j\pi m^2\Delta_t\Delta_f} \left(e^{j\pi n^2\Delta_t\Delta_f} * e^{-j\pi n^2\Delta_t\Delta_f} g(n\Delta_t) \right)$$

convolution

Z-transform: $X(z)=\sum_{n=0}^{N-1}x[n]z^{-n} \longrightarrow X(k)=X(z)\Big|_{z=e^{j\frac{2\pi k}{N}}}$

CZT algorithm:

Define $Z_k = AW^{-k}$, k=0, 1, …, M-1, 其中M為任意output points
A和W為任意complex number。

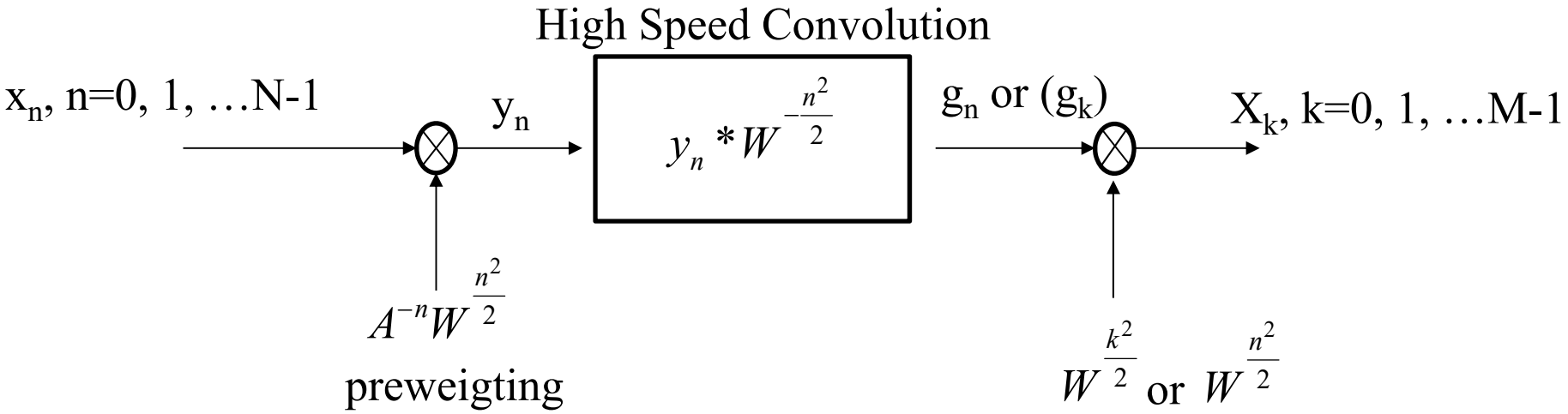
$$X_k=\sum_{n=0}^{N-1}x[n](\mathrm{AW}^{-k})^{-n}=\sum_{n=0}^{N-1}x[n]\mathrm{A}^{-n}\mathrm{W}^{kn},\ k=0,1,...,M-1$$

令 $nk=\frac{n^2+k^2-(\mathrm{k}-\mathrm{n})^2}{2}$ 代入並整理得:

$$X_k=\sum_{n=0}^{N-1}(\underbrace{x[n]\mathrm{A}^{-n}\mathrm{W}^{\frac{n^2}{2}}}_{\mathrm{y_n}})\mathrm{W}^{\frac{k^2}{2}}\underbrace{\mathrm{W}^{\frac{-(\mathrm{k}-n)^2}{2}}}_{\mathrm{v_{k-n}}},\ k=0,1,...,M-1$$

$$\Rightarrow X_k=\mathrm{W}^{\frac{k^2}{2}}\sum_{n=0}^{N-1}\mathrm{y}[n]v[k-n]=\mathrm{W}^{\frac{k^2}{2}}\left(y[\mathrm{k}]*\mathrm{v}[\mathrm{k}]\right),\ k=0,1,...,M-1$$

Block diagram:



優點:

(1)input/output point 可以不相同($N \neq M$)，N和M為任意整數

(2)contour 不需要在單位圓上(arc即可)

(3)初始點任意(arbitrary initial frequency)，而DFT必須要DC點開始

缺點: 運算量較大 (3 times)

◉ 10-H Winograd Algorithm for DFT Implementation

Basic idea:

Except for the 1st row and the 1st column, the *N*-point DFT is equivalent to the (*N*−1)-point circular convolution when *N* is a prime number.

Example: 5-point DFT

$$\begin{bmatrix} V_0 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 \\ 1 & \omega^2 & \omega^4 & \omega & \omega^3 \\ 1 & \omega^3 & \omega & \omega^4 & \omega^2 \\ 1 & \omega^4 & \omega^3 & \omega^2 & \omega \end{bmatrix} \begin{bmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}, \qquad \omega = \exp[-j\angle 72°],$$

移除第一個 row和第一個 column

$$\begin{bmatrix} V_1 - v_0 \\ V_2 - v_0 \\ V_3 - v_0 \\ V_4 - v_0 \end{bmatrix} = \begin{bmatrix} \omega & \omega^2 & \omega^3 & \omega^4 \\ \omega^2 & \omega^4 & \omega & \omega^3 \\ \omega^3 & \omega & \omega^4 & \omega^2 \\ \omega^4 & \omega^3 & \omega^2 & \omega \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}$$

先將 3rd and 4th rows, 再2nd, 3rd and 4th columns 作交換

$$\begin{bmatrix} V_1 - v_0 \\ V_2 - v_0 \\ V_4 - v_0 \\ V_3 - v_0 \end{bmatrix} = \begin{bmatrix} \omega & \omega^2 & \omega^4 & \omega^3 \\ \omega^3 & \omega & \omega^2 & \omega^4 \\ \omega^4 & \omega^3 & \omega & \omega^2 \\ \omega^2 & \omega^4 & \omega^3 & \omega \end{bmatrix} \begin{bmatrix} v_1 \\ v_3 \\ v_4 \\ v_2 \end{bmatrix}$$

變成 circular convolution 的型態

Circular Convolution

$$z[n] = y[n] \otimes h[n] = \sum_{k=0}^{N-1} y[k] h[((n-k))_N]$$

$$\longrightarrow \quad z[n] = IFFT\{FFT(y[n]) FFT(h[n])\}$$

$$\begin{bmatrix} V_1 - v_0 \\ V_2 - v_0 \\ V_4 - v_0 \\ V_3 - v_0 \end{bmatrix} = IFFT\left(FFT_4\left\{\begin{bmatrix} v_1 \\ v_3 \\ v_4 \\ v_2 \end{bmatrix}\right\} \cdot FFT_4\left\{\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_4 \\ \omega_3 \end{bmatrix}\right\}\right)$$

↑ independent of input

$$FFT_4\left\{\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_4 \\ \omega_3 \end{bmatrix}\right\} = \begin{bmatrix} -1 \\ -1.7156 - 1.9021j \\ 2.2361 \\ 1.7156 - 1.9021j \end{bmatrix}$$

1 N-point DFT
= 2 (N-1)-point DFTs
+ N-1 pixel-wise multiplications

當 N 為其他的 prime numbers 時，也可以運用 permutation 和 circular convolution來計算 prime-number DFTs

(Step 1) Delete the 1st row and the 1st column.

(Step 2) Perform the row and column permutations.

Rows 和 columns 的順序相同

(a) 找出一個 primitive root a, 使得 $a^k \bmod N \neq 1$ when $k = 1, 2, \ldots, N-2$,

$a^{N-1} \bmod N \neq 1$ （Primitive root 的概念，會在後面講到數論時複習）

(b) Rows 和 columns 的順序，以 $p[n]$ 來表示，

$p[n] = a^n \bmod N, \quad n = 0, 1, \ldots, N-2$

(Step 3) 變成 circular convolution 的型態

則 N-point DFT 可以用 $(N-1)$-point DFTs 來 implementation

$$\begin{bmatrix} V_{p[0]} - v_0 \\ V_{p[1]} - v_0 \\ \vdots \\ V_{p[N-2]} - v_0 \end{bmatrix} = IDFT_{N-1}\left(DFT_{N-1}\left\{ \begin{bmatrix} v_{p[0]} \\ v_{p[N-2]} \\ \vdots \\ v_{p[1]} \end{bmatrix} \right\} DFT_{N-1}\left\{ \begin{bmatrix} w^{p[0]} \\ w^{p[1]} \\ \vdots \\ w^{p[N-2]} \end{bmatrix} \right\} \right)$$

重要理論：

Any N-point DFT can be implemented by the 2^k-point DFTs whatever the value of N is.

7-point DFT

123-point DFT