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

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

(1)

(2)

(3)

(4)

◉ 9-B 對於簡單矩陣快速演算法的設計

如何簡化下面四個運算

(1) $y[1] = ax[1] + 2ax[2]$

(2) $\begin{bmatrix} y(1) \\ y(2) \end{bmatrix} = \begin{bmatrix} a & a \\ a & a \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \end{bmatrix}$

(3) $\begin{bmatrix} y(1) \\ y(2) \end{bmatrix} = \begin{bmatrix} a & b \\ b & a \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \end{bmatrix}$

(4) $\begin{bmatrix} y(1) \\ y(2) \end{bmatrix} = \begin{bmatrix} a & b \\ c & a \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \end{bmatrix}$

(4)

$$\begin{bmatrix} y(1) \\ y(2) \end{bmatrix} = \begin{bmatrix} a & b \\ c & a \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \end{bmatrix} = \begin{bmatrix} a & a \\ a & a \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \end{bmatrix} + \begin{bmatrix} 0 & b-a \\ c-a & 0 \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \end{bmatrix}$$

$$\begin{bmatrix} z(1) \\ z(2) \end{bmatrix} = \begin{bmatrix} a & a \\ a & a \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \end{bmatrix} \qquad \begin{bmatrix} z(3) \\ z(4) \end{bmatrix} = \begin{bmatrix} 0 & b-a \\ c-a & 0 \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \end{bmatrix}$$

(i) $z(1) = a[x(1) + x(2)], \quad z(2) = z(1)$

(ii) $z(3) = (b-a)x(2), \qquad z(4) = (c-a)x(1)$

(iii) $y(1) = z(1) + z(3), \qquad y(2) = z(2) + z(4)$

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

◉ 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

$$\begin{bmatrix} y[1] \\ y[2] \\ y[3] \\ y[4] \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & f & e \\ f & e & e & f \\ d & c & b & a \end{bmatrix} \begin{bmatrix} x[1] \\ x[2] \\ x[3] \\ x[4] \end{bmatrix}$$

◉ 9-D Examples

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

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

- 5×5 DFT 的例子

real part **imaginary part**

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & a & b & b & a \\ 1 & b & a & a & b \\ 1 & b & a & a & b \\ 1 & a & b & b & a \end{bmatrix} - j \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & c & d & -d & -c \\ 0 & d & -c & c & -d \\ 0 & -d & c & -c & d \\ 0 & -c & -d & d & c \end{bmatrix}$$

$$a = \cos(2\pi/5)$$

$$b = \cos(4\pi/5)$$

$$c = \sin(2\pi/5)$$

$$d = \sin(4\pi/5)$$

8-point DCT

$$\begin{bmatrix} y[0] \\ y[1] \\ y[2] \\ y[3] \\ y[4] \\ y[5] \\ y[6] \\ y[7] \end{bmatrix} = \begin{bmatrix} 0.7010 & 0.7010 & 0.7010 & 0.7010 & 0.7010 & 0.7010 & 0.7010 & 0.7010 \\ 0.9808 & 0.8315 & 0.5556 & 0.1951 & -0.1951 & -0.5556 & -0.8315 & -0.9808 \\ 0.9239 & 0.3827 & -0.3827 & -0.9239 & -0.9239 & -0.3827 & 0.3827 & 0.9239 \\ 0.8315 & -0.1951 & -0.9808 & -0.5556 & 0.5556 & 0.9808 & 0.1951 & -0.8315 \\ 0.7010 & -0.7010 & -0.7010 & 0.7010 & 0.7010 & -0.7010 & -0.7010 & 0.7010 \\ 0.5556 & -0.9808 & 0.1951 & 0.8315 & -0.8315 & -0.1951 & 0.9808 & -0.5556 \\ 0.3827 & -0.9239 & 0.9239 & -0.3827 & -0.3827 & 0.9239 & -0.9239 & 0.3827 \\ 0.1951 & -0.5556 & 0.8315 & -0.9808 & 0.9808 & -0.8315 & 0.5556 & -0.1951 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \\ x[6] \\ x[7] \end{bmatrix}$$

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:
$$\begin{bmatrix} y[0] \\ y[2] \\ y[4] \\ y[6] \end{bmatrix} = \begin{bmatrix} 0.7010 & 0.7010 & 0.7010 & 0.7010 \\ 0.9239 & 0.3827 & -0.3827 & -0.9239 \\ 0.7010 & -0.7010 & -0.7010 & 0.7010 \\ 0.3827 & -0.9239 & 0.9239 & -0.3827 \end{bmatrix} \begin{bmatrix} z[0] \\ z[1] \\ z[2] \\ z[3] \end{bmatrix}$$

Part 2: $$\begin{bmatrix} y[1] \\ y[3] \\ y[5] \\ y[7] \end{bmatrix} = \begin{bmatrix} 0.9808 & 0.8315 & 0.5556 & 0.1951 \\ 0.8315 & -0.1951 & -0.9808 & -0.5556 \\ 0.5556 & -0.9808 & 0.1951 & 0.8315 \\ 0.1951 & -0.5556 & 0.8315 & -0.9808 \end{bmatrix} \begin{bmatrix} z[4] \\ z[5] \\ z[6] \\ z[7] \end{bmatrix}$$

[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: $O((N_x N_y)\log_2(N_x N_y))$ Why?
- Convolution of an M-point sequence and an N-point sequence:

$O((M+N-1)\log_2(M+N-1))$ 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,

$O(M_xM_y)$ when $M_xM_y >> N_xN_y$

$O(N_xN_y)$ 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 $N\log_2 N$

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)

(2) Radix-4, 8, 16, …. Algorithms

(3) Prime Factor Algorithm

(4) Goertzel Algorithm

(5) Chirp Z transform (CZT)

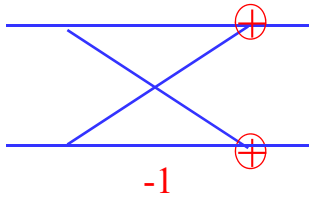
(6) Winograd algorithm

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.

◉ 10-B Cooley Tukey Algorithm

When $N = 2^k$



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

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

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

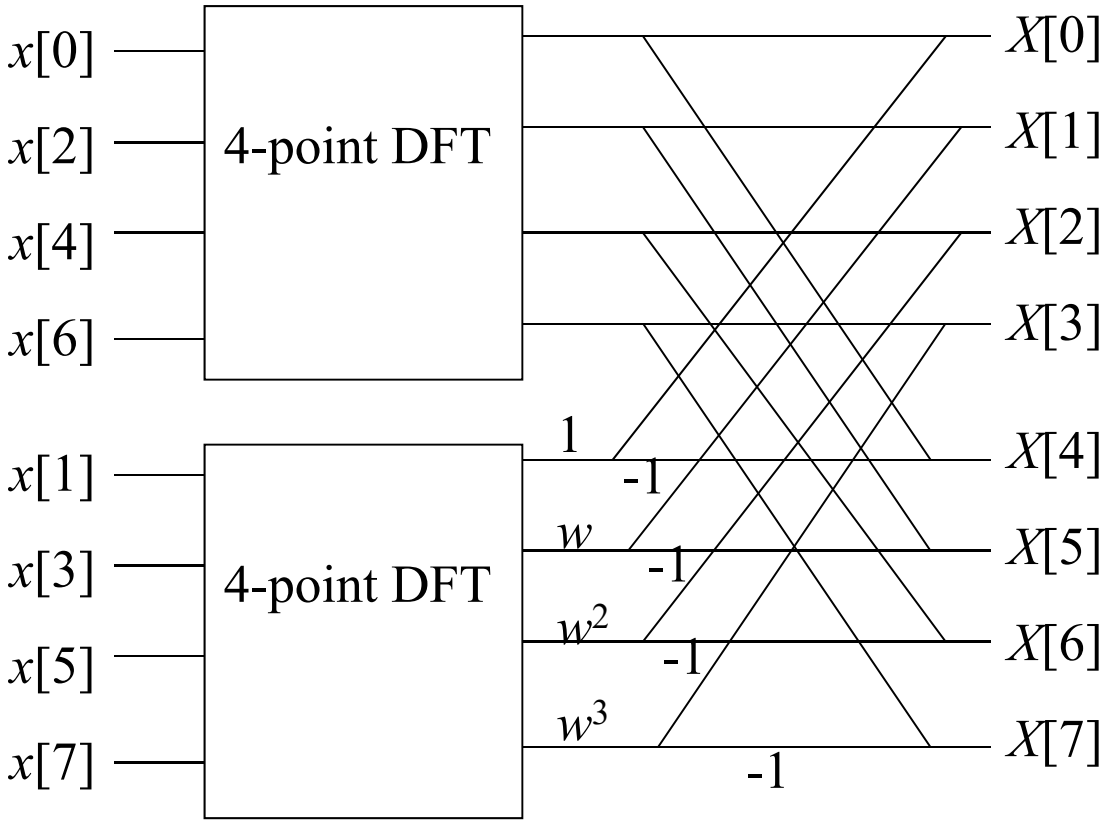
twiddle factors

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

Therefore,

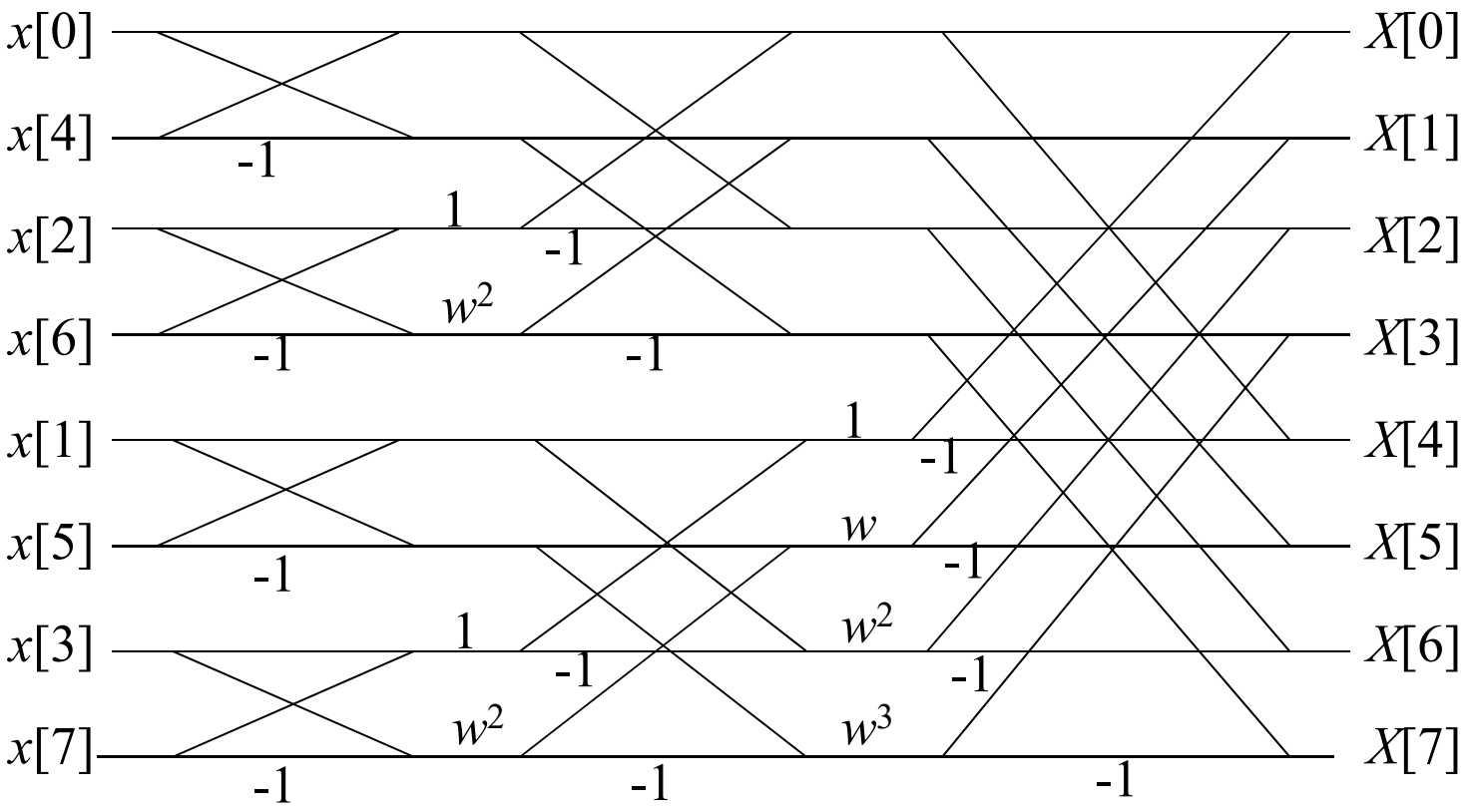
one N-point DFT = two $(N/2)$-point DFT + twiddle factors

8-point DFT



$$w = e^{-j\frac{2\pi}{8}} \qquad w^4 = -1 \qquad w^8 = 1$$

8-point DFT



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

- 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

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

∴ 2^k-point DFT 需要

$$32^{k-1}(k-1)=\frac{3}{2}N\left(\log_2 N-1\right)$$ 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)

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

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

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

twiddle factors

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

N	乘法數	加法數	N	乘法數	N	乘法數	N	乘法數
1	0	0	11	46	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	160	54	228
9	16	72	21	62	35	150	56	156
10	20	88	22	80	36	64	60	160

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
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	412	168	580	312	1324	784	4412
90	340	180	680	336	1412	840	4580

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

附錄十：論文英文常見的文法錯誤

(1) *** transform, *** equation, *** method, *** algorithm 在論文當中，當成是可數名詞，而非專有名詞 (除非是所有格的形態)。

可數名詞單數時，前面要要冠詞 (a 或 the)

Fourier transform is important for signal processing. （錯誤）

The Fourier transform is important for signal processing. （正確）

A Fourier transform is important for signal processing. （正確）

Fourier transforms are important for signal processing. （正確）

I have written the Matlab program of Parks-McClellan algorithm （錯誤）

I have written the Matlab program of the Parks-McClellan algorithm （正確）

(2) 若是所有格的形態，不必加冠詞

I have written the Matlab program of the Parks-McClellan's algorithm （錯誤）

I have written the Matlab program of Parks-McClellan's algorithm （正確）

(3) 論文視同正式的文件，對 not, is, are 不用縮寫

they’re（錯誤）　they are（正確）

he’s（錯誤）　he is（正確）

aren’t（錯誤）　are not（正確）

don’t（錯誤）　do not（正確）

can’t（錯誤）　cannot（正確）

(4) Suppose, assume 後面要加關係代名詞

Suppose *x* is a large number.（錯誤）

Suppose that *x* is a large number.（正確）

(5) 每一個子句都有一個動詞，而且只有一個動詞

(6) In this paper, in this section, in this chapter 開頭的句子，應該用現在式，而非未來式

In this paper, the fast algorithm of DCT will be introduced. （錯誤）

In this paper, , the fast algorithm of DCT is introduced. （正確）

(7) 在 conclusion 當中回顧文章一內容，用過去式

(8) 敘述所引用的論文的內容，用過去式

In [10], the number theoretic transform was proposed.

(9) time domain, frequency domain 前面也加冠詞

in time domain （錯誤） in the time domain （正確）

(10) 不以 “this paper”, “section *”, “Ref. [*]” 當主詞用

This paper describes several concepts. （錯誤）

In this paper, several concepts are described. （正確）

Ref. [1] proposed the method. （錯誤）

In Ref. [1], Parks and McClellan proposed the method. （正確）

(11) 提及某個 equation 時，直接括號加數字即可

in equation (3)（錯誤） in (3)（正確）

提及某個 section, table, or figure 時，前面不加冠詞，而且常用大寫

in the section 4（錯誤） in Section 4（正確）

in the table 5（錯誤） in Table 4（正確）

(12) 寫科技論文不是寫文學作品，不要用高明、漂亮、但沒有保握的文法。

儘量用簡單而有把握的文法。

(13) 科技論文英文講求「長話短說」，儘量用精簡的文字來表達意思

(14) 用字儘量避免重覆

(15) Equations 也當成是文章的一部分，所以通常也要加標點符號

The formula of Newton's 2nd law is

$$F = ma.$$

要加標點符號

(16) 解釋 parameters 和 symbols 時，用 where 當關係代名詞

$x = 10t$ where x is the location of the object and t is time.

(17) 很重要的論文，投稿至國際學術期刊，又對自己的英文文法沒有十足的把握時

可以用網路上的論文編修服務，來修改文法上的錯誤

本系以及台大語言中心也經常有英文論文寫作相關的訓練課程，有志將來在學術界奮鬥的同學，可以多參與相關的課程