VII. Data Compression (A)

♦**壓縮的通則：**

利用資料的一致性

資料越一致的資料，越能夠進行壓縮

[References]

- I. Bocharova, *Compression for Multimedia*, Cambridge, UK, Cambridge University Press, 2010.
- 酒井善則，吉田俊之原著，原島博監修，白執善編譯，“影像壓縮術”，全華印行, 2004.
- 戴顯權，“資料壓縮 Data Compression,” 旗標出版社, 2007.
- D. Salomon, *Introduction to Data Compression*, Springer, 3rd ed., New York, 2004.

◉ 7-A 壓縮的哲學：

(1) 利用資料的一致性，規則性，與可預測性

(exploit redundancies and predictability, find the compact or sparse representation)

(2) 通常而言，若可以用比較精簡的自然語言來描述一個東西，那麼也就越能夠對這個東西作壓縮

Q: 最古老的壓縮技術是什麼？

(3) 資料越一致，代表統計特性越集中

包括 Fourier transform domain, histogram, eigenvalue ……….. 等方面的集中度

Data type	**Compression technique**	**Compression rate**
Audio		
Image		
Video		

思考：如何對以下的資料作壓縮

Article:

Song:

Voice:

Cartoon:

Compression: Original signal →Compact representation + residual information

◎ 7-B Compression for Images

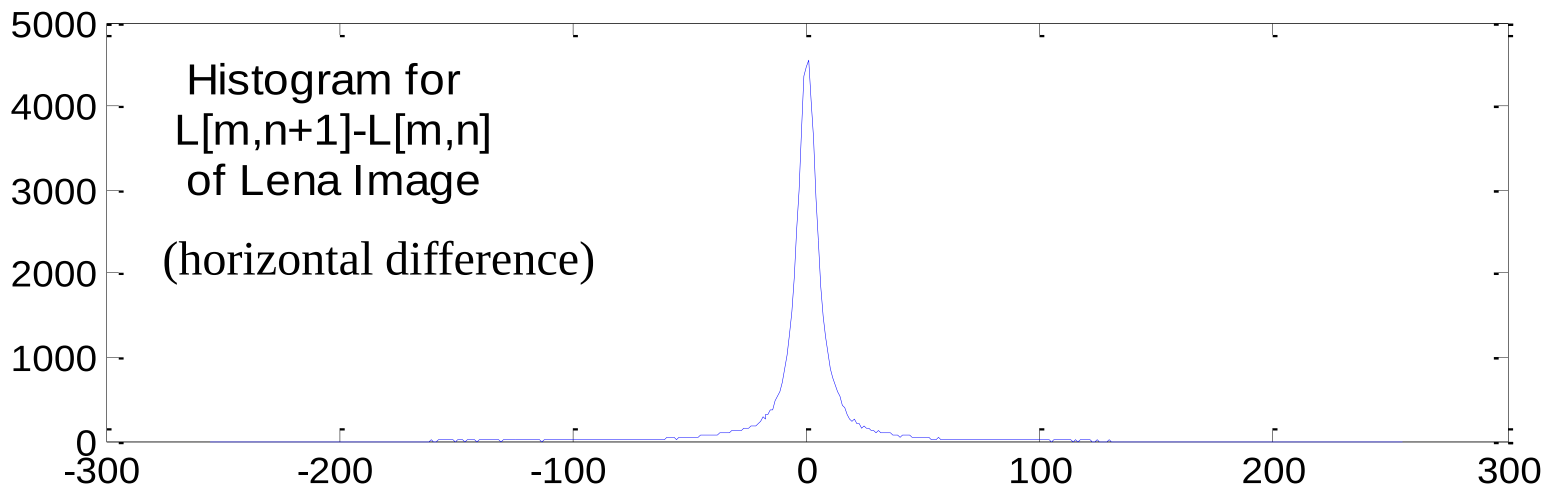
- **影像的「一致性」：**

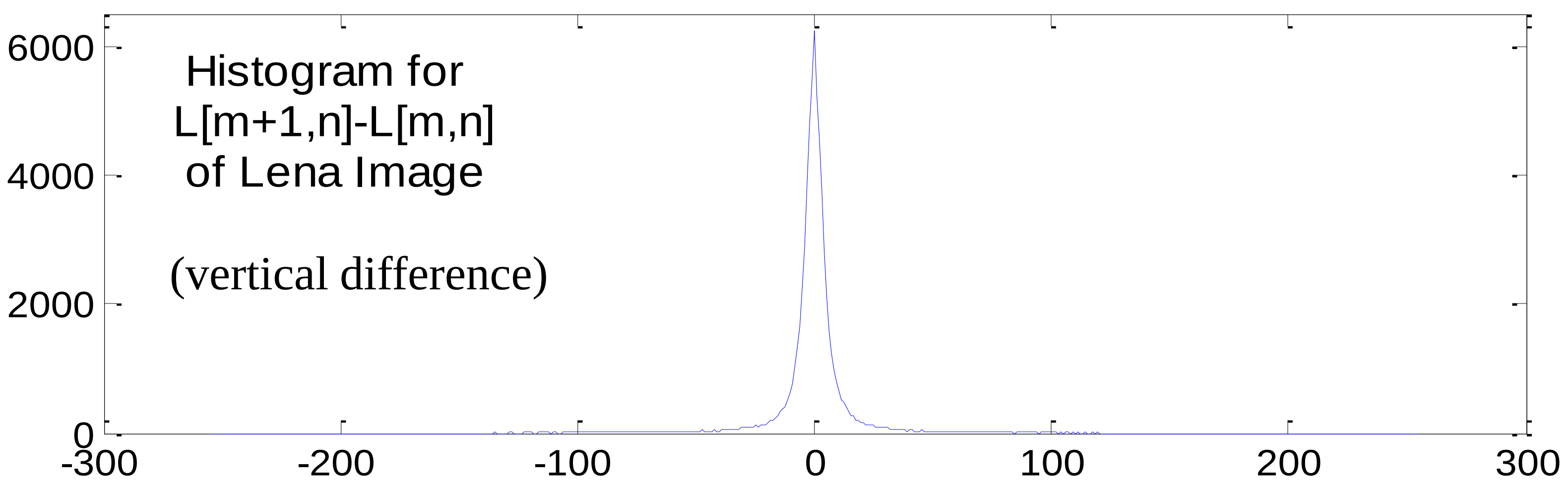
Space domain: 每一點的值，會和相鄰的點的值非常接近

$$F[m, n+1] \approx F[m, n], \qquad F[m+1, n] \approx F[m, n]$$

Frequency domain: 大多集中在 **低頻** 的地方。

Lena Image 在 space domain 上的一致性





Histogram:

一個 vector 或一個 matrix 當中，有多少點會等於某一個值

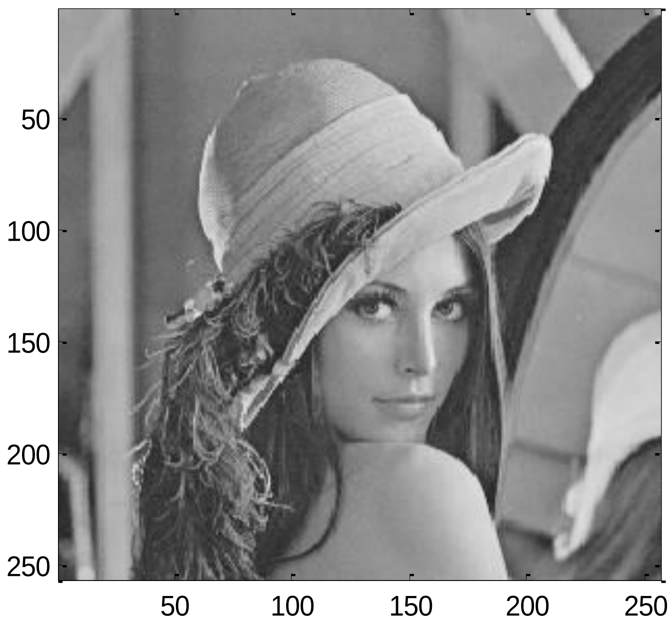
例如：$x[n] = [1\ 2\ 3\ 4\ 4\ 5\ 5\ 3\ 5\ 5\ 4]$

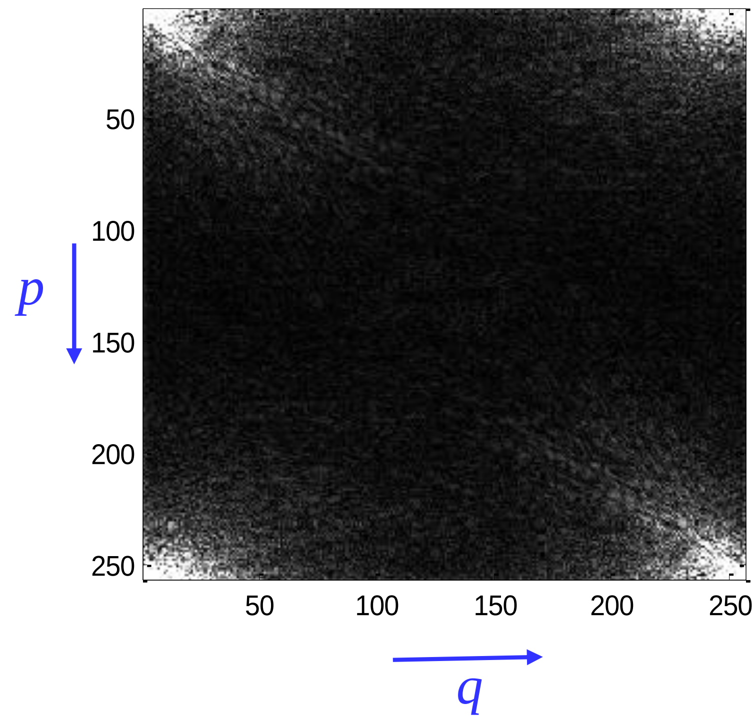
則 $x[n]$ 的 histogram 為

$h[1] = 1$, $h[2] = 1$, $h[3] = 2$, $h[4] = 3$, $h[5] = 4$

Lena Image 頻譜 (frequency domain) 的一致性

$L[m, n]$ |fft2($L[m, n]$)| (用亮度來代表 amplitude)





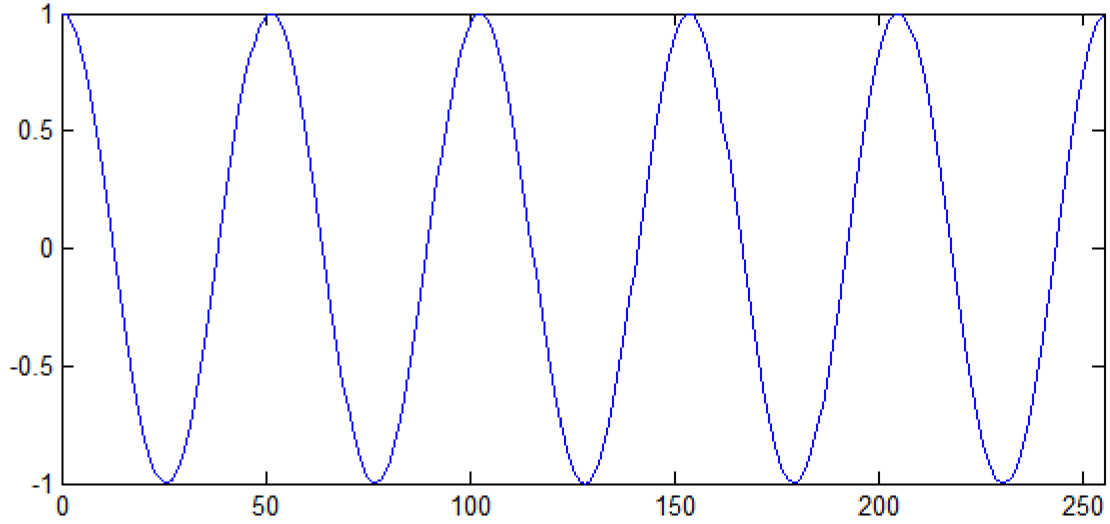
$$L_F[p,q] = fft2\{L[m,n]\} = \sum_{m=1}^{M}\sum_{n=1}^{N} L[m,n]\, e^{-j2\pi\frac{pm}{M}} e^{-j2\pi\frac{qn}{N}}$$

$$L[m,n] = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N} L_F[p,q]\, e^{j2\pi\frac{pm}{M}} e^{j2\pi\frac{qn}{N}}$$

影像的「頻率」：frequency in the space domain

$e^{j2\pi\frac{pm}{M}}$ 從 $m = 0$ 至 $m = M$-1 之間有 p 個週期

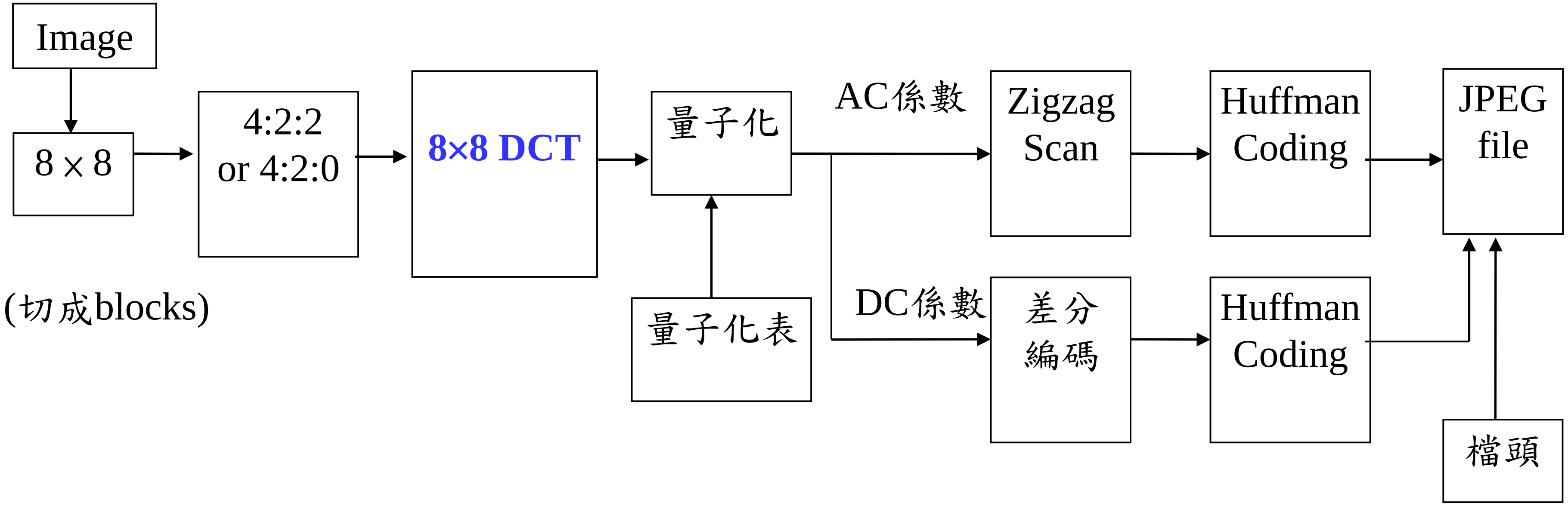
$p = 5$ $\mathrm{Re}\{e^{j2\pi\frac{pm}{M}}\}$



larger p： more variation in the space domain

◉ 7.C JPEG Standard

Process of JPEG Image Compression



• 主要用到四個技術：(1) 4:2:2 or 4:2:0 (和 space domain 的一致性相關)

(2) 8 × 8 DCT (和 frequency domain 的一致性相關)

(3) 差分編碼 (和 space domain 的一致性相關)

(4) Huffman coding (和 lossless 編碼技術相關)

JPEG： 影像編碼的國際標準　　全名： Joint Photographic Experts Group

JPEG 官方網站： http://www.jpeg.org/

參考論文： G. K. Wallace, “The JPEG still picture compression standard,” *IEEE Transactions on Consumer Electronics*, vol. 38, issue 1, pp. 18-34, 1992.

JPEG 的 FAQ 網站： http://www.faqs.org/faqs/jpeg-faq/

JPEG 的免費 C 語言程式碼：
http://opensource.apple.com/source/WebCore/WebCore-1C25/platform/image-decoders/jpeg/

一般的彩色影像，可以壓縮 12~20 倍。

簡單的影像甚至可以壓縮超過 20 倍。

- **壓縮的技術分成兩種**

lossy compression techniques

無法完全重建原來的資料

Examples: DFT, DCT, KLT (with quantization and truncation), 4:2:2 or 4:2:0, polynomial approximation

壓縮率較高

lossless compression techniques

可以完全重建原來的資料

Examples: binary coding, Huffman coding, arithmetic coding, Golomb coding

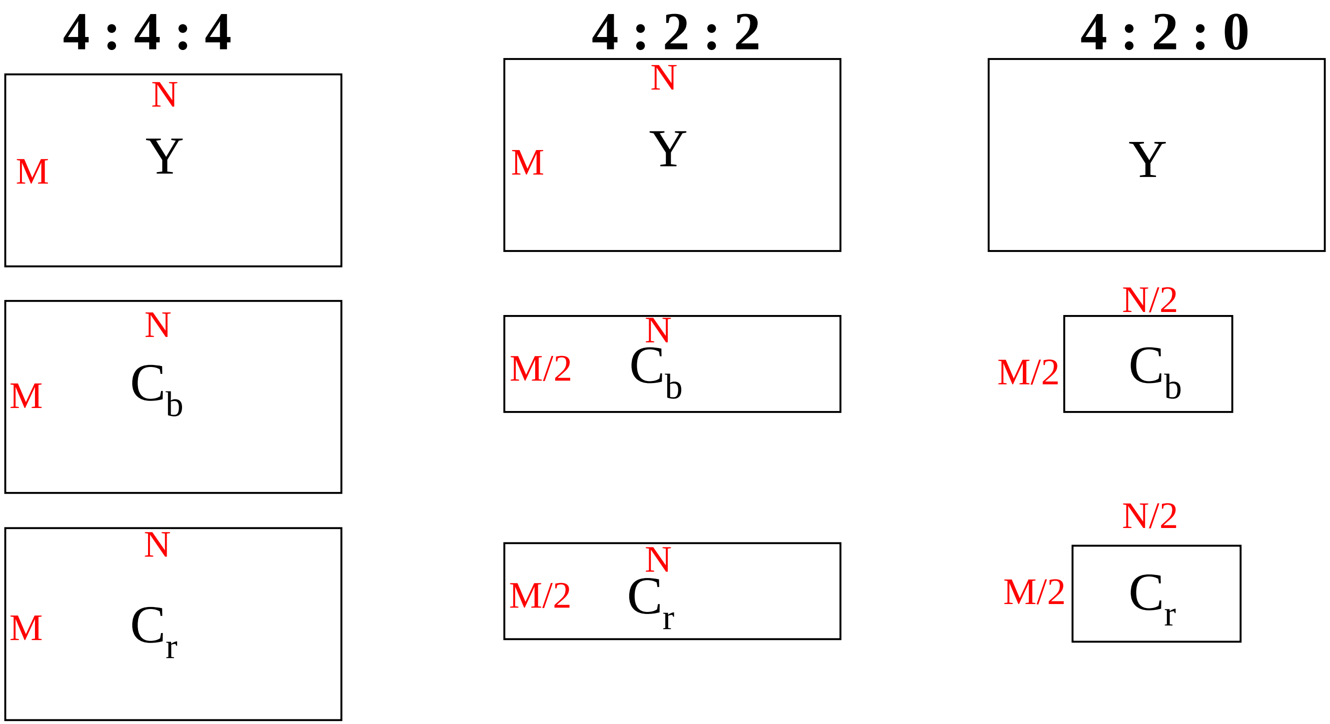
壓縮率較低

◉ 7-D 4:2:2 and 4:2:0

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

R: red, G: green, B: blue

Y: 亮度, C_b: $0.565(B-Y)$, C_r: $0.713(R-Y)$,

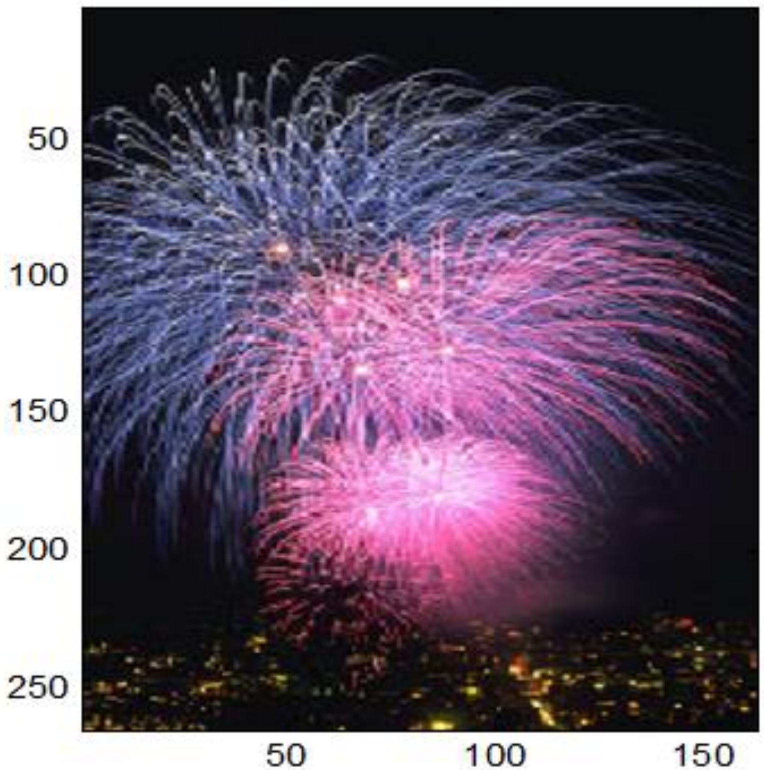


24 bits/pixel → 16 bits/pixel → 12 bits/pixel

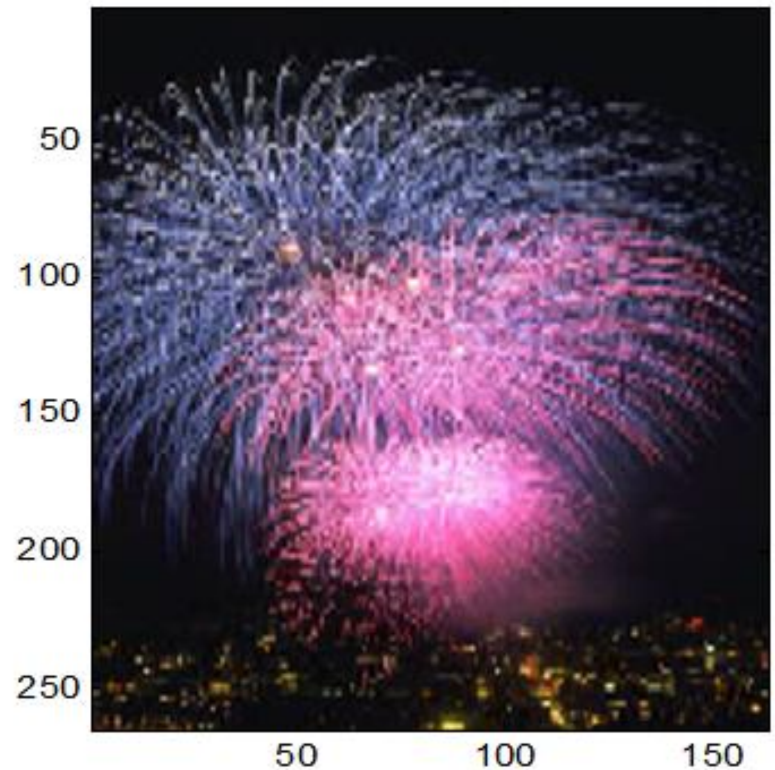
同樣使資料量省一半的(b)(d)圖，(d)圖和原來差不多，
然而(b)圖邊緣會有失真現象。

還原時，用 interpolation 的方式

原圖　　　　直接在縱軸取一半的pixels 再還原

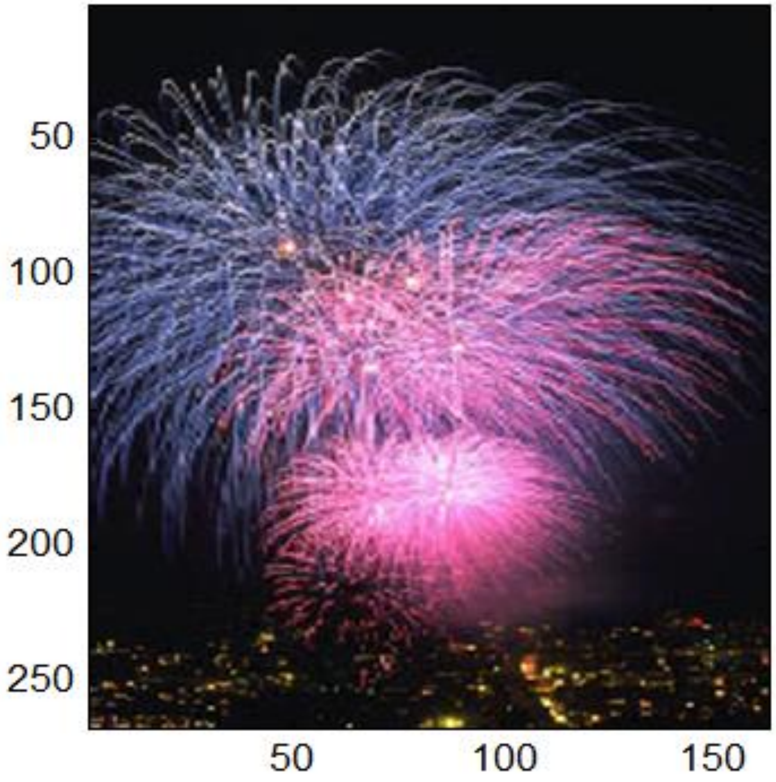


(a)



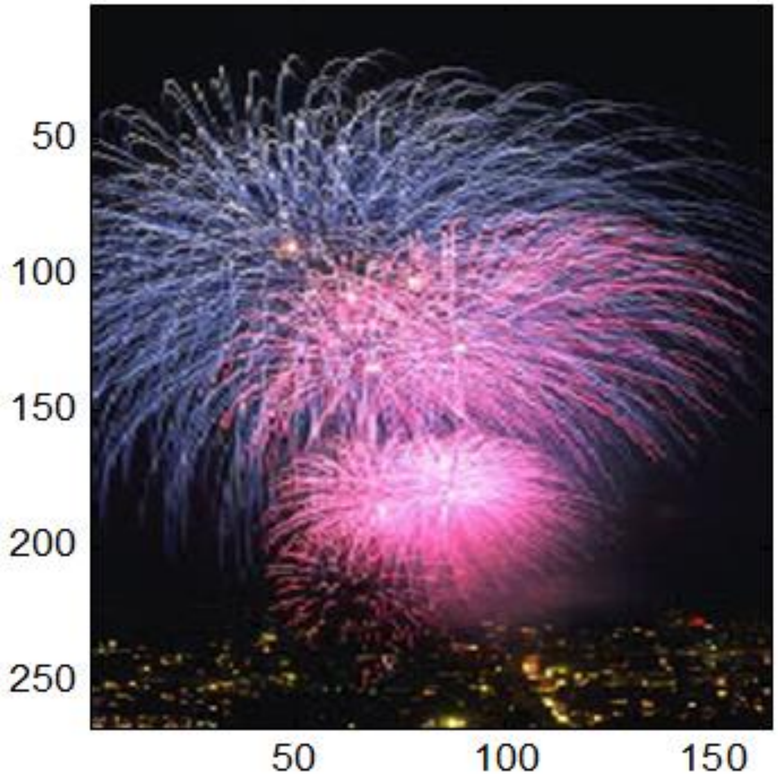
(b)

4 : 2: 2



(c)

4 : 2: 0



(d)

◉ 7-E Lossy Compression Techniques -- KLT

複習：DFT 的優缺點

It is optimal, but dependent on the input

- **Karhunen-Loeve Transform (KLT)**

(similar to Principal component analysis (PCA))

經過轉換後，能夠將影像的能量分佈變得最為集中

分析影像的主要成分，第二主要成份，第三主要成份，…………

- 1-D Case $X[u] = \sum_{n=0}^{N-1} x[n] K[u,n]$

$K[u, n] = e_n[u]$ (K = [e_0, e_1, e_2, ….., e_{N-1}])

$\mathbf{e_n}$ 為 covariance matrix **C** 的 eigenvector

mean

$C[m, n] = \mathrm{corr}(x[m], x[n]) = E\left[(x[m]-\overline{x[m]})(x[n]-\overline{x[n]})\right]$

Note: corr代表correlation

KLT 的理論基礎：

經過 KLT 之後，當 $u_1 \neq u_2$ 時，$X[u_1]$ 和 $X[u_2]$ 之間的 correlation 必需近於零 (即 decorrelation)

即 $\text{corr}(X[u_1],\ X[u_2]) = E\left[(X[u_1]-\overline{X[u_1]})(X[u_2]-\overline{X[u_2]})\right]=0$

所以 $E\left[X[u_1]X[u_2]\right]-\overline{X[u_1]}\ \overline{X[u_2]}=0$

Since if $\overline{x[n]}=0$ $\quad \overline{X[u]}=\sum_{n=0}^{N-1}\overline{x[n]}K[u,n]=0$ for all u

The above equation can be simplified as:

$$E\left[X[u_1]X[u_2]\right]=0$$

$$E\left[X[u_1]X[u_2]\right]=0$$

Note that $E\left[X[u_1]X[u_2]\right]$ is the $(u_1, u_2)^{th}$ entry of $E\{\mathbf{XX^T}\}$

where $\mathbf{X}=\left[X[0], X[1], X[2], \cdots, X[N-1]\right]^T$

Since $\mathbf{X}=\mathbf{Kx}$ where $\mathbf{x}=\left[x[0], x[1], x[2], \cdots, x[N-1]\right]^T$

K is the KLT matrix

$$E\left\{\mathbf{XX^T}\right\}=E\left\{\mathbf{Kxx^TK^T}\right\}=\mathbf{K}E\left\{\mathbf{xx^T}\right\}\mathbf{K^T}=\mathbf{KCK^T}$$

where **C** is the covariance matrix and

$$\text{corr}(x[m], x[n]) = E\left[(x[m]-\overline{x[m]})(x[n]-\overline{x[n]})\right]=E\left[x[m]x[n]\right]$$

To make $E\left[X[u_1]X[u_2]\right]=0$ when $u_1 \neq u_2$

$E\left\{\mathbf{XX^T}\right\}$ should be a diagonal matrix

Therefore, the KLT transform matrix **K** should diagonalize **C**.

That is, the columns of **K** are the eigenvectors of **C**.

- 2-D Case $$X[u,v]=\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}x[m,n]K[u,m]K[v,n]$$

KLT 缺點: dependent on image
(不實際，需要一併記錄 transform matrix)

Reference

W. D. Ray and R. M. Driver, “Further decomposition of the Karhunen-Loeve series representation of a stationary random process,” *IEEE Trans. Inf. Theory*, vol. 16, no. 6, pp. 663-668, Nov. 1970.

◉ 7-F Lossy Compression Techniques -- DCT

Suboptimal, but independent of the input

- **DCT: Discrete Cosine Transform**

$$F[u,v]=\frac{2C[u]C[v]}{N}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}f[m,n]\cos\frac{(2m+1)u\pi}{2N}\cos\frac{(2n+1)v\pi}{2N}$$

$$C[0]=1/\sqrt{2}\quad, C[u]=1 \text{ for } u\neq 0$$

IDCT: inverse discrete cosine transform

$$f[m,n]=\frac{2}{N}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}F[u,v]C[u]C[v]\cos\frac{(2m+1)u\pi}{2N}\cos\frac{(2n+1)v\pi}{2N}$$

對於大部分的影像而言， DCT 能夠近似 KLT (near optimal)

尤其是當 corr$\{f[m, n], f[m+\tau, n+\eta]\} = \rho^{|\tau|}\rho^{|\eta|}$, $\rho\to 1$ 時

有 fast algorithm

Advantage: (1) independent of the input (2) near optimal (3) real output

DCT

$$F[u,v]=\frac{2C[u]C[v]}{N}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f[m,n]\cos\frac{(2m+1)u\pi}{2N}\cos\frac{(2n+1)v\pi}{2N}$$

$$C[0]=1/\sqrt{2} \qquad , C[u]=1 \text{ for } u\neq 0$$

$[u, v] = [0, 0]$: DC term

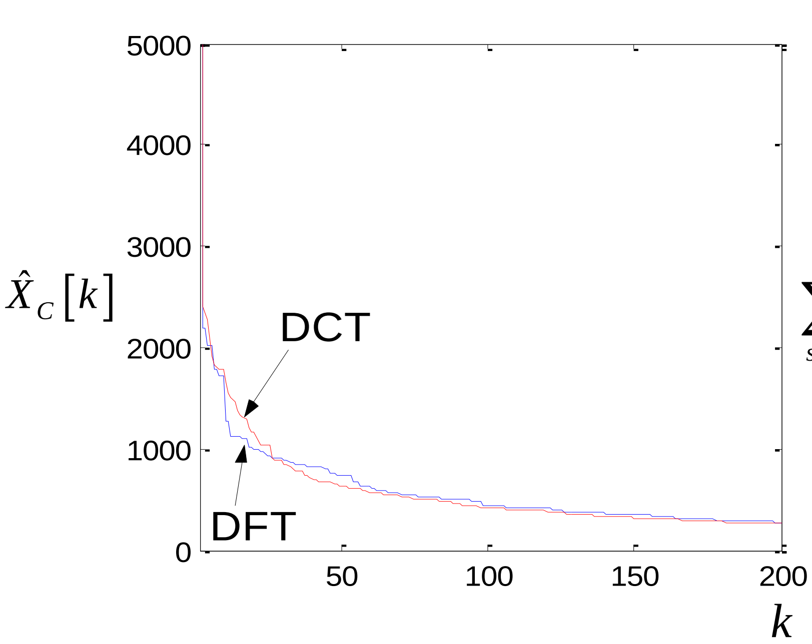
$u \neq 0$ or $v \neq 0$: AC terms

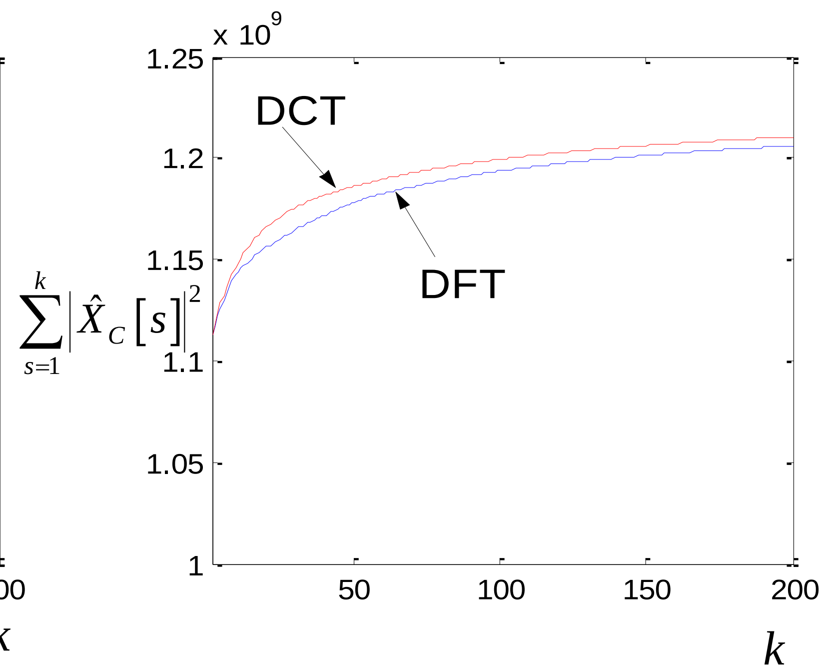
借用電路學的名詞

左圖：將 DFT，DCT 各點能量(開根號)由大到小排序
右圖：累積能量

DCT output

$$X_C[p,q] \xrightarrow{sort} \hat{X}_C[k] \qquad \hat{X}_C[1] \geq \hat{X}_C[2] \geq \hat{X}_C[3] \geq \cdots\cdots$$

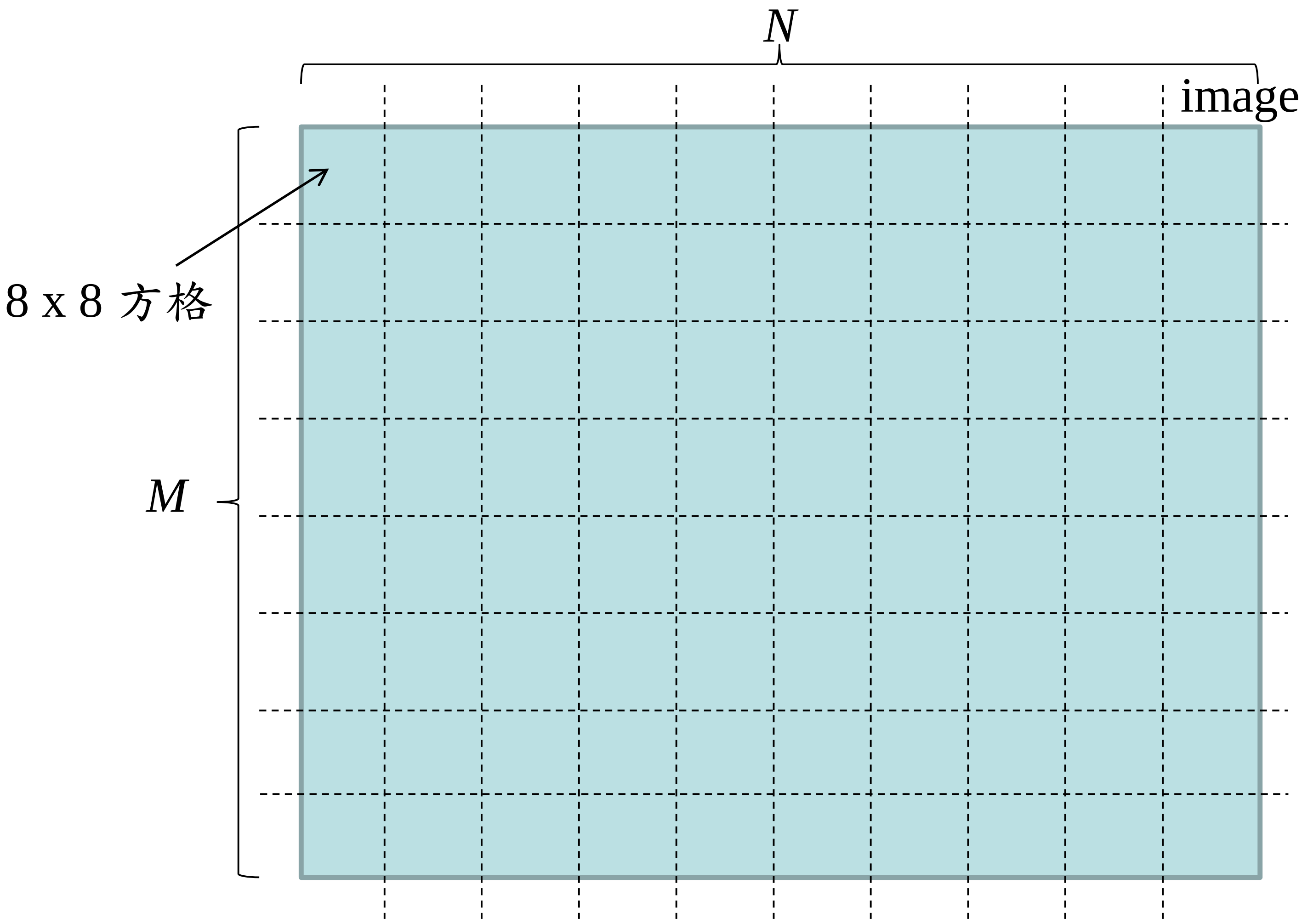




Energy concentration at low frequencies: KLT > DCT > DFT

通常，我們將影像切成 8×8 的方格作DCT

Why:



References

[1] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, pp. 90-93, Jan 1974.

[2] K. R. Rao and P. Yip, *Discrete Cosine Transform, Algorithms, Advantage, Applications*, New York: Academic, 1990.

VIII. Data Compression (B)

◉ 8-A Differential Coding for DC Terms, Zigzag for AC Terms

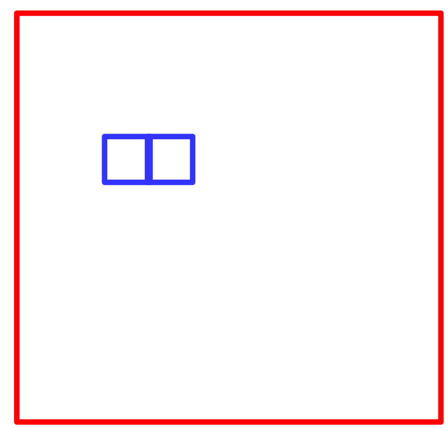
這兩者可視為 JPEG Huffman coding 的前置工作

Differential Coding (差分編碼)

If the DC term of the $(i,\ j)^{\text{th}}$ block is denoted by $DC[i, j]$, then

encode $DC[i, j] - DC[i, j\text{-}1]$

Instead of $DC[i, j]$



(也是運用 space domain 上的一致性)

Zigzag scanning

將 2D 的 8x8 DCT outputs 變成 1D 的型態

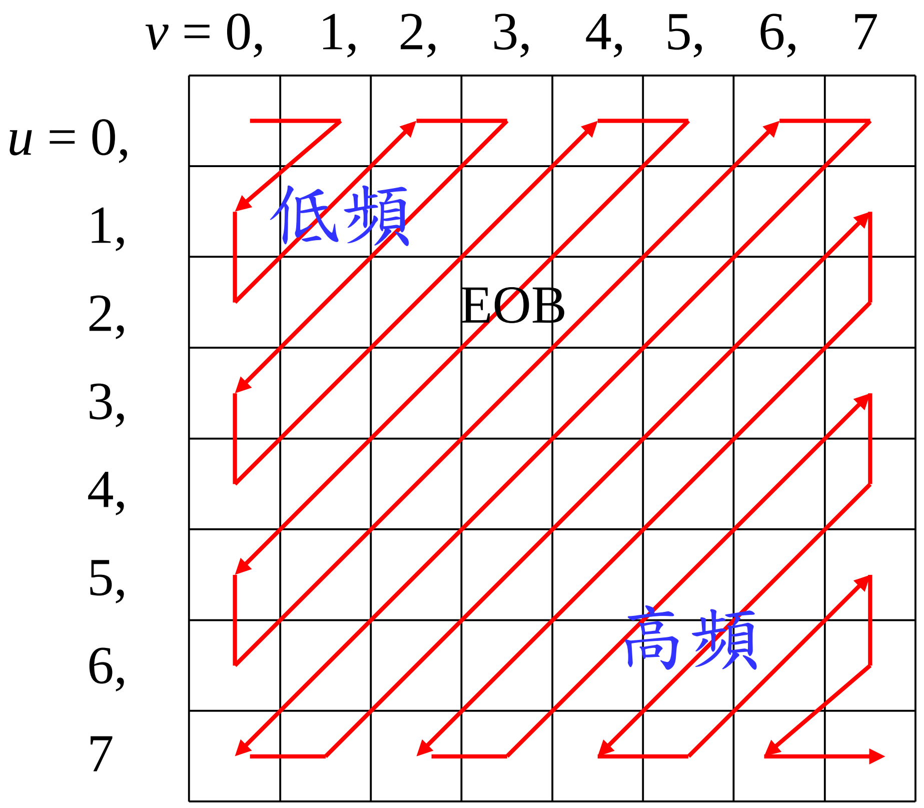
但按照 “zigzag” 的順序 (能量可能較大的在前面)

8x8 DCT output:

$C[u, v]$

$u = 0, 1, \ldots, 7$

$v = 0, 1, \ldots, 7$



EOB (end of block):

指後面的高頻的部分經過 quantization 之後皆為0

(也是運用 frequency domain 上的一致性)

◉ 8-B Lossless Coding

Lossless Coding: The original data can be perfectly recovered

Example:

direct coding method

Huffman coding

Arithmetic coding

Shannon–Fano Coding, Golomb coding, Lempel–Ziv, ...

◉ 8-C Lossless Coding: Huffman Coding

- **Huffman Coding 的編碼原則: (Greedy Algorithm)**

(1) 所有的碼皆在 Coding Tree 的端點，再下去沒有分枝
(滿足一致解碼和瞬間解碼)

(2) 機率越大的， code length 越短；機率越小的， code length 越長

(3)假設 S_a 是第 L 層的 node，S_b是第 L+1 層的 node

則 $P(S_a) \geq P(S_b)$ 必需滿足

不滿足以上的條件則交換

原始的編碼方式：

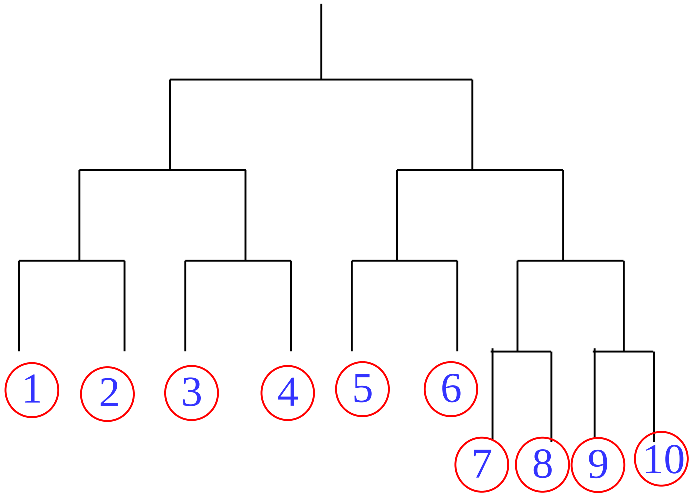
若 data 有 M 個可能的值，使用 k 進位的編碼，
則每一個可能的值使用 floor($\log_k M$) 或 ceil($\log_k M$) 個 bits 來編碼

floor: 無條件捨去

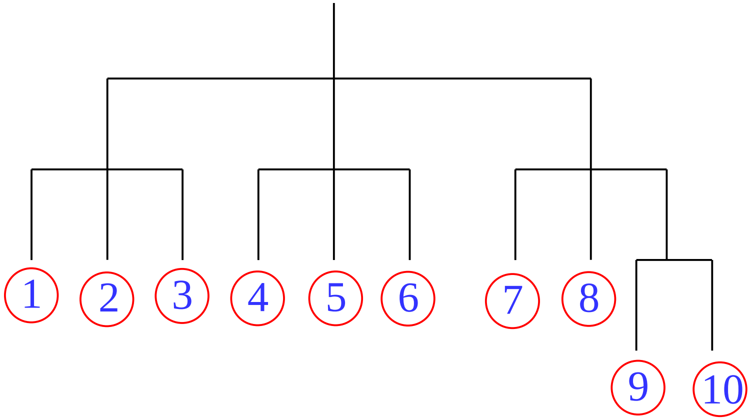
ceil: 無條件進位

Example:

若有 8 個可能的值，在2進位的情形下，需要 3 個 bits



若有 10 個可能的值，在3進位的情形下，需要 2 個或 3 個 bits



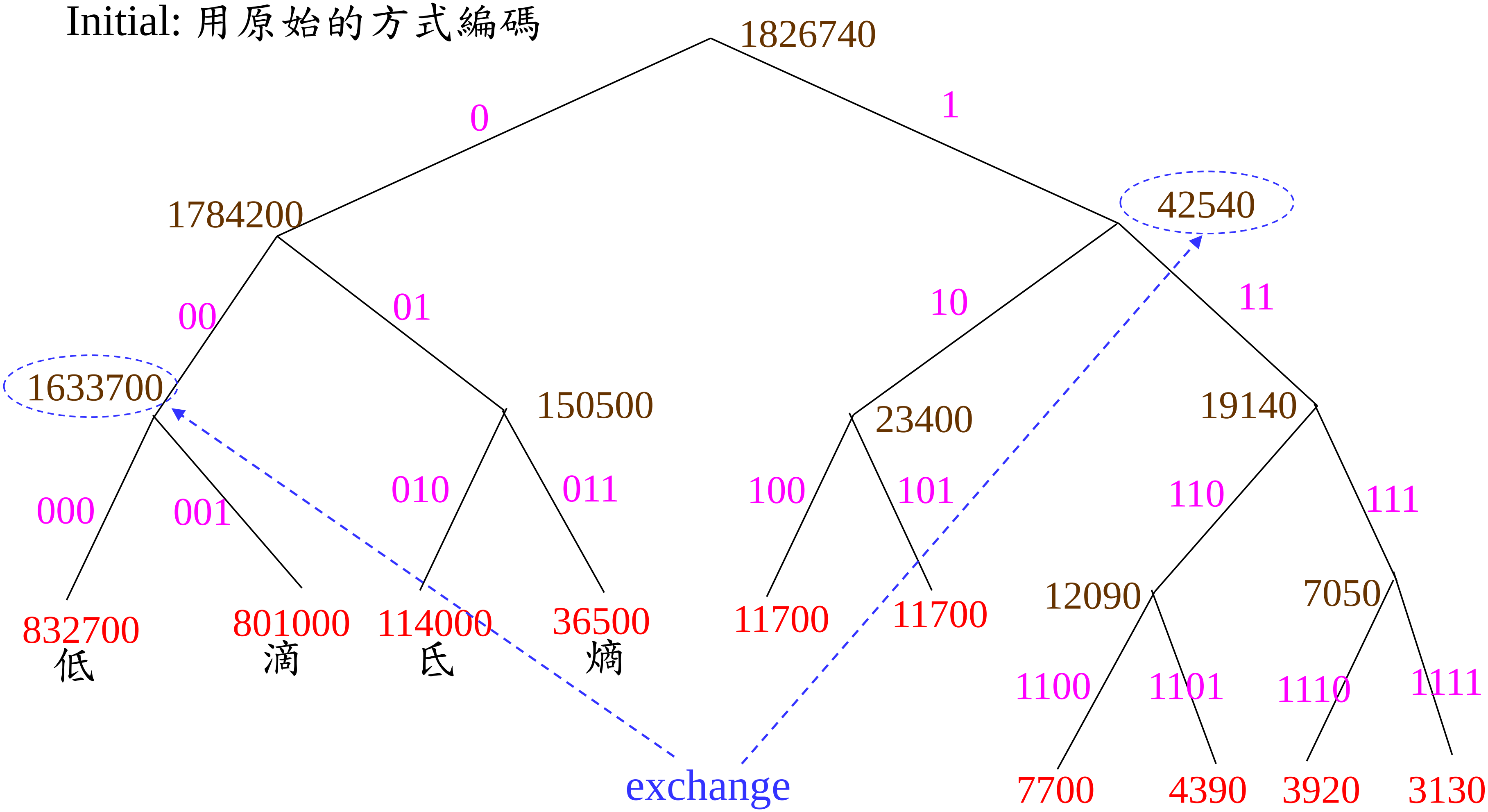
Example:

低	滴	氐	羝	鞮
832700	801000	114000	7700	4390
磾	祇	菂	墑	熵
3920	11700	11700	3130	36500

他們 2進位的Huffman Code 該如何編

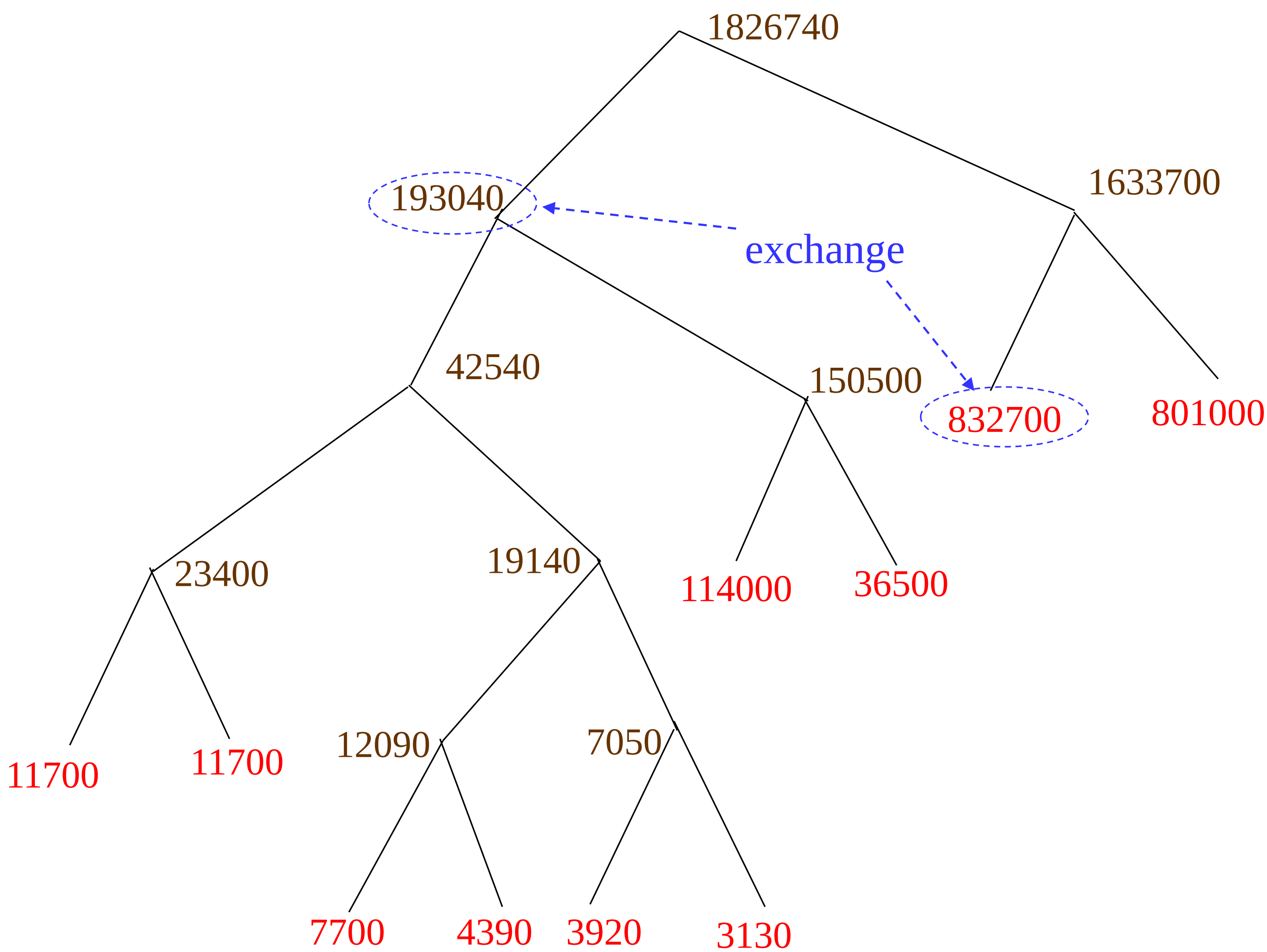
Huffman coding by the greedy algorithm

Initial: 用原始的方式編碼

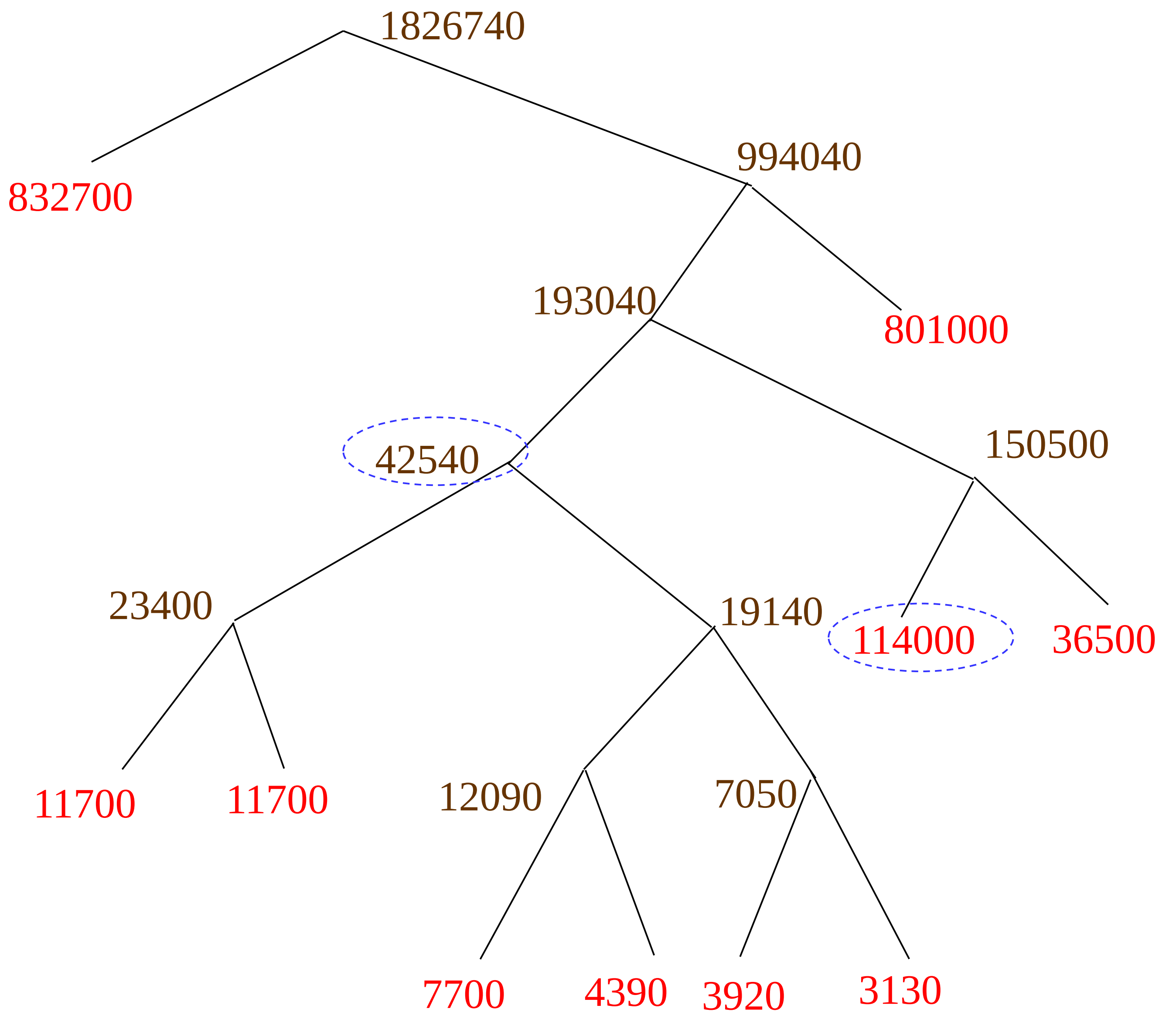


average code length = 3.0105

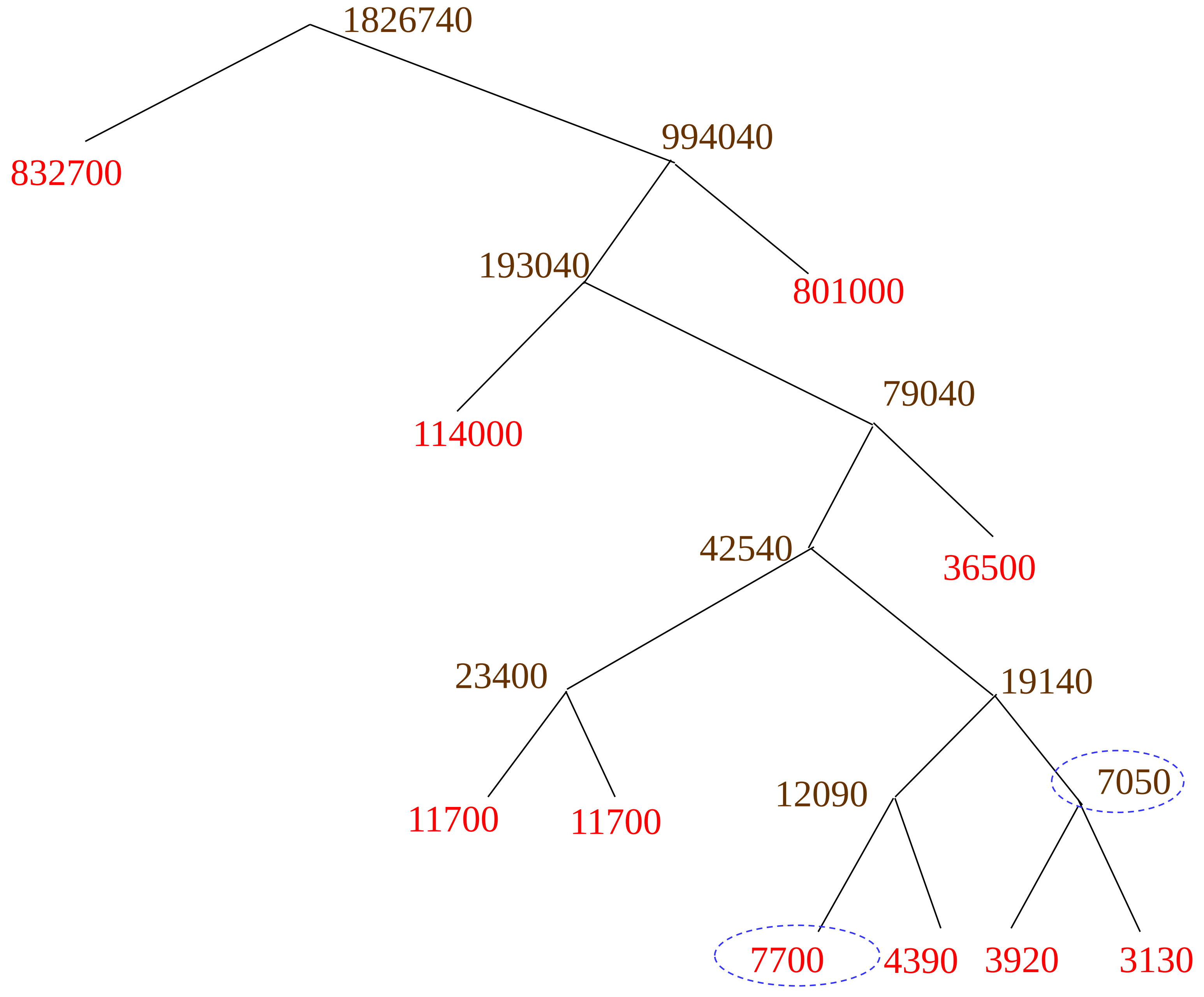
Huffman coding by the greedy algorithm



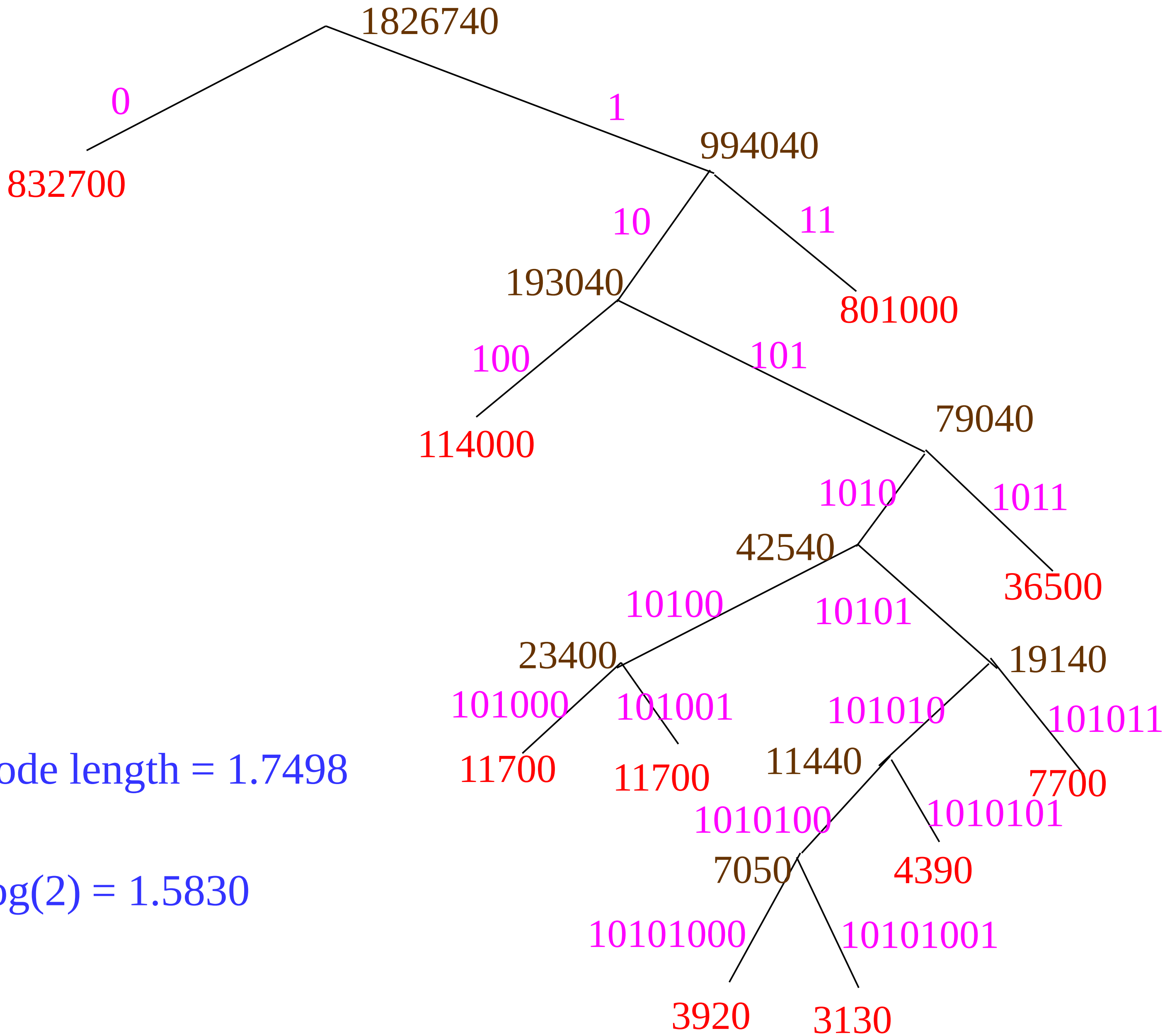
Huffman coding by the greedy algorithm



Huffman coding by the greedy algorithm



Huffman coding by the greedy algorithm



average code length = 1.7498

entropy/log(2) = 1.5830

思考： 郵遞區號是多少進位的編碼？

電話號碼的區域碼是多少進位的編碼？

中文輸入法是多少進位的編碼？

如何用 Huffman coding 來處理類似問題？

◉ 8-D Entropy and Coding Length

- Entropy 熵；亂度 (Information Theory)

註：此處 log 即 ln 和 $\log_{10}$ 不同

$$entropy = \sum_{j=1}^{J} P\left(S_j\right)\log\frac{1}{P\left(S_j\right)}$$

P: probability

$P(S_0) = 1$, entropy = 0

$P(S_0) = P(S_1) = 0.5$, entropy = 0.6931

$P(S_0) = P(S_1) = P(S_2) = P(S_3) = P(S_4) = 1/5$, entropy = 1.6094

$P(S_0) = P(S_1) = P(S_2) = P(S_3) = 0.1$, $P(S_4) = 0.6$, entropy = 1.2275

同樣是有 5 種組合，機率分佈越集中，亂度越少

- Huffman Coding 的平均長度

$$mean(L)=\sum_{j=1}^{J}P(S_j)L(S_j)$$

$P(S_j)$: S_j 發生的機率， $L(S_j)$: S_j 的編碼長度

- **Shannon 編碼定理：**

$$\frac{entropy}{\log k}\le mean(L)\le\frac{entropy}{\log k}+1$$

若使用 k 進位的編碼

- Huffman Coding 的 total coding length $b=mean(L)N$ N: data length

$$\boxed{ceil\left(N\frac{entropy}{\log k}\right)\le b\le floor\left(N\frac{entropy}{\log k}+N\right)}$$

都和 entropy 有密切關係

ceil: 無條件進位, floor: 無條件捨去

Entropy: 估計 coding length 的重要工具

$$N \frac{entropy}{\log k} \cong \text{bit length}$$

◉ 8-E Arithmetic Coding

- **Arithmetic Coding (算術編碼)**

Huffman coding 是將每一筆資料分開編碼

Arithmetic coding 則是將多筆資料一起編碼，因此壓縮效率比 Huffman coding 更高，近年來的資料壓縮技術大多使用 arithmetic coding

K. Sayood, *Introduction to Data Compression*, Chapter 4: Arithmetic coding, 3rd ed., Amsterdam, Elsevier, 2006

編碼

若 data X 有 M 個可能的值 ($X[i]$ = 1, 2, …, or M)，使用 k 進位的編碼，且

P_n : the probability of $x = n$ (from prediction)

$$S_0 = 0, \quad S_n = \sum_{j=1}^{n} P_j$$

現在要對 data X 做編碼，假設 length(X) = N

Algorithm for arithmetic encoding

initiation: $lower = S_{X[1]-1}$ $upper = S_{X[1]}$

for i = 2 : N

$$lower = lower + S_{X[i]-1} \times (upper - lower)$$

$$upper = lower + S_{X[i]} \times (upper - lower)$$

end

(continue)…

Suppose that

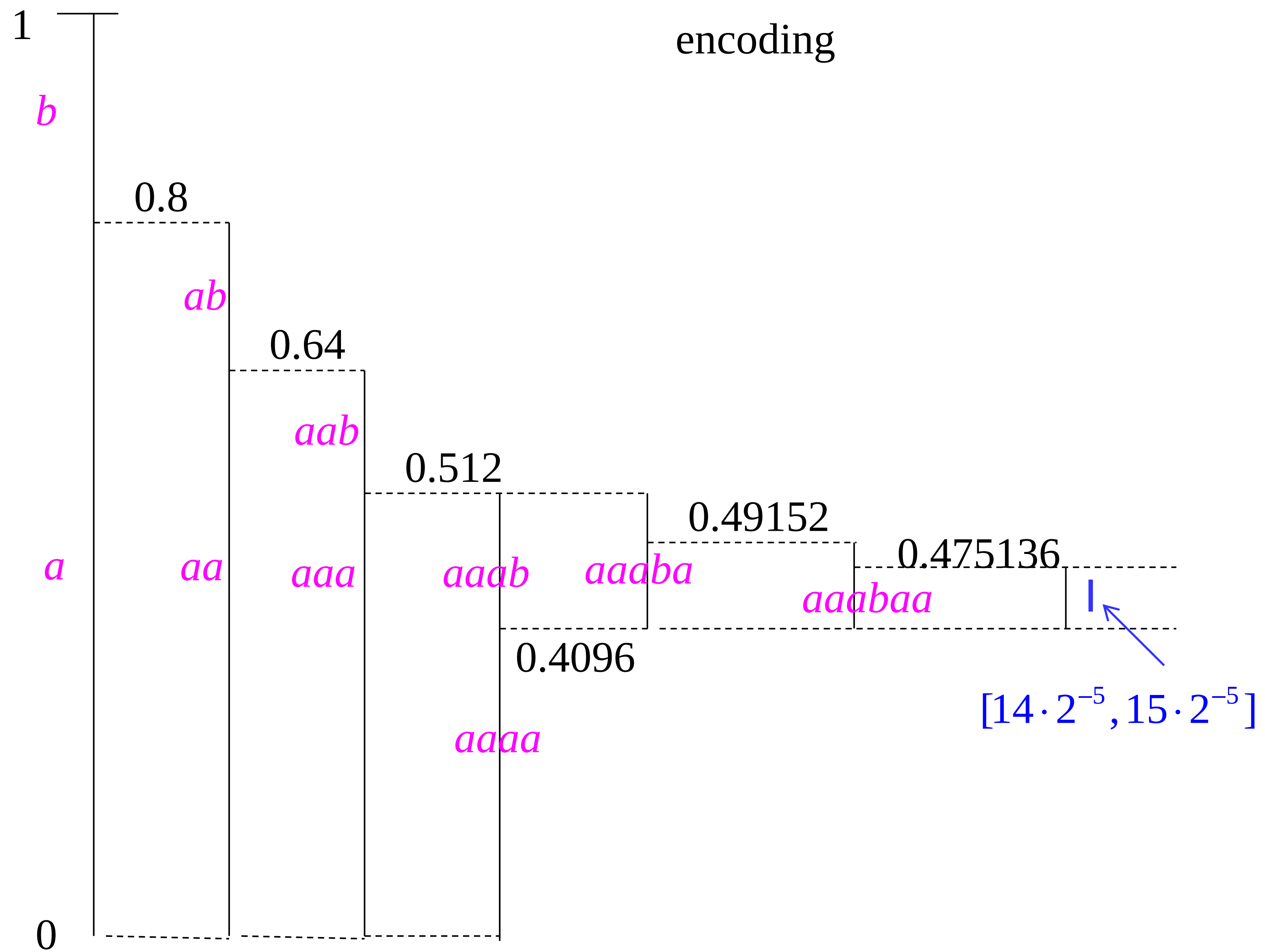
$$lower \le C \cdot k^{-b} < (C+1) \cdot k^{-b} \le upper$$

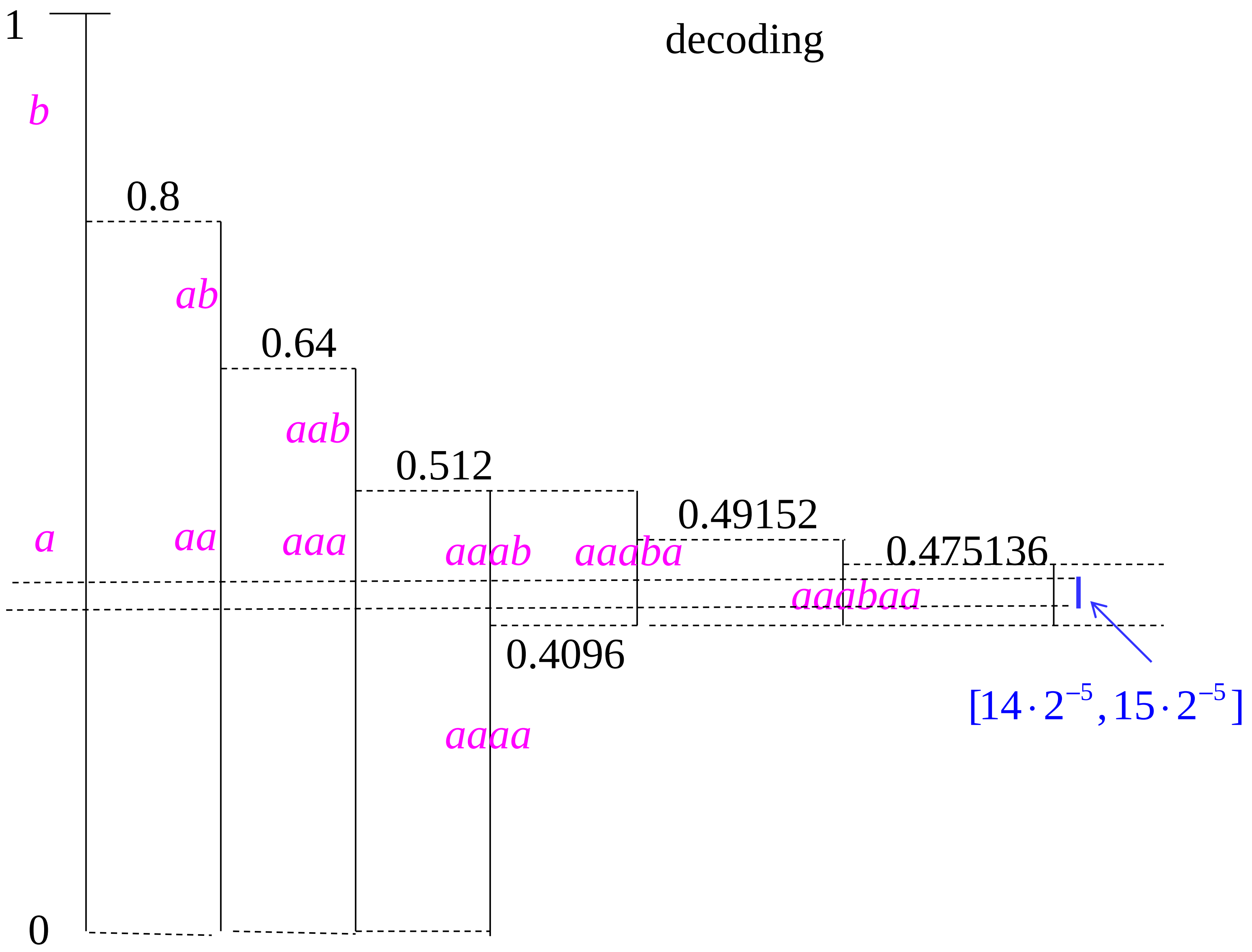
where C and b are integers (b is as small as possible), then the data X can be encoded by

$$C_{(k,b)}$$

where $C_{(k,b)}$ means that using k-ary (k 進位) and b bits to express C.

(註： Arithmetic coding 還有其他不同的方式，以上是使用其中一個較簡單的 range encoding 的方式)


1
encoding
b
0.8
ab
0.64
aab
0.512
0.49152
0.475136
a
aa
aaa
aaab
aaaba
aaabaa
I
0.4096
$[14 \cdot 2^{-5}, 15 \cdot 2^{-5}]$
aaaa
0


decoding
1
b
0.8
ab
0.64
aab
0.512
0.49152
0.475136
a
aa
aaa
aaab
aaaba
aaabaa
0.4096
[14·2^{-5}, 15·2^{-5}]
aaaa
0

Example:

假設要對 X 來做二進位 ($k = 2$) 的編碼

且經由事先的估計，$X[i] = a$ 的機率為 0.8, $X[i] = b$ 的機率為 0.2

$$\Rightarrow \quad P_1 = 0.8, \quad P_2 = 0.2, \qquad S_0 = 0, \quad S_1 = 0.8, \quad S_2 = 1$$

若實際上輸入的資料為 $X = a\ a\ a\ b\ a\ a$

Initiation ($X[1] = a$): $lower = 0, \quad upper = 0.8$

When $i = 2$ ($X[2] = a$): $lower = 0, \quad upper = 0.64$

When $i = 3$ ($X[3] = a$): $lower = 0, \quad upper = 0.512$

When $i = 4$ ($X[4] = b$): $lower = 0.4096, \quad upper = 0.512$

When $i = 5$ ($X[5] = a$): $lower = 0.4096, \quad upper = 0.49152$

When $i = 6$ ($X[6] = a$): $lower = 0.4096, \quad upper = 0.475136$

由於 $lower = 0.4096, \quad upper = 0.475136$

$$lower \le 14 \cdot 2^{-5} < 15 \cdot 2^{-5} \le upper$$

0.4375 0.46875

所以編碼的結果為

$$14_{(2,5)} = 01110$$

2 進位 5 個 bits

解碼

假設編碼的結果為 Y, length(Y) = b

其他的假設，和編碼 (see page 283) 相同　　使用 k 進位的編碼

Algorithm for arithmetic decoding

initiation: $lower = 0$　　$upper = 1$　　$j = 1$

$lower\,1 = 0$　　$upper\,1 = 1$

for $i = 1 : N$　　% loop 1

check = 1;

while check = 1　　% loop 2

if there exists an n such that

$lower + (upper\text{-}lower)S_{n-1} \leqq$ lower 1　　and

$lower + (upper\text{-}lower)S_{n} \geqq$ upper 1　　are both satisfied,

then

check = 0;

(continue)….

else

$$upper1 = lower1 + (Y[j]+1)k^{-j}$$

$$lower1 = lower1 + Y[j]k^{-j}$$

$j = j+1$

end

end % end of loop 2

$X(i) = n;$

$lower = lower + (upper\text{-}lower)S_{n\text{-}1}$

$upper = lower + (upper\text{-}lower)S_n$

end % end of loop 1

Coding Length for Arithmetic Coding

假設 P_n 是預測的 $X[i] = n$ 的機率

Q_n 是實際上的 $X[i] = n$ 的機率

(也就是說，若 length(X) = N, X 當中會有 $Q_n N$ 個 elements 等於 n)

則

$$upper - lower = \prod_{m=1}^{M} P_m^{Q_m N}$$

$\prod$: 連乘符號

另一方面，由於 (from page 284)

$$k^{-b} \le upper - lower < (2k)k^{-b}$$

$$-\log_k (upper - lower) \le b < -\log_k (upper - lower) + 1 + \log_k 2$$

$$ceil\left(-N\sum_{m=1}^{M} Q_m \log_k P_m\right) \le b \le floor\left(-N\sum_{m=1}^{M} Q_m \log_k P_m + \log_k 2\right) + 1$$

$$ceil\left(-N\sum_{m=1}^{M}Q_m\log_k P_m\right)\le b\le floor\left(-N\sum_{m=1}^{M}Q_m\log_k P_m+\log_k 2\right)+1$$

在機率的預測完全準確的情形下， $Q_m = P_m$

Total coding length b 的範圍是

$$ceil\left(-N\sum_{m=1}^{M}P_m\log_k P_m\right)\le b\le floor\left(-N\sum_{m=1}^{M}P_m\log_k P_m+\log_k 2\right)+1$$

$$\boxed{ceil\left(N\cdot\frac{entropy}{\log k}\right)\le b\le floor\left(N\cdot\frac{entropy}{\log k}+\log_k 2+1\right)}$$

Arithmetic coding 的 total coding length 的上限比 Huffman coding 更低

◎ 8-F MPEG

MPEG： 動態影像編碼的國際標準　　全名： Moving Picture Experts Group

MPEG standard： http://www.iso.org/iso/prods-services/popstds/mpeg.html

MPEG 官方網站： http://mpeg.chiariglione.org/

人類的視覺暫留： 1/24 second

一個動態影像，每秒有 30個或 60個畫格 (frames)

例子：

Pepsi 的廣告

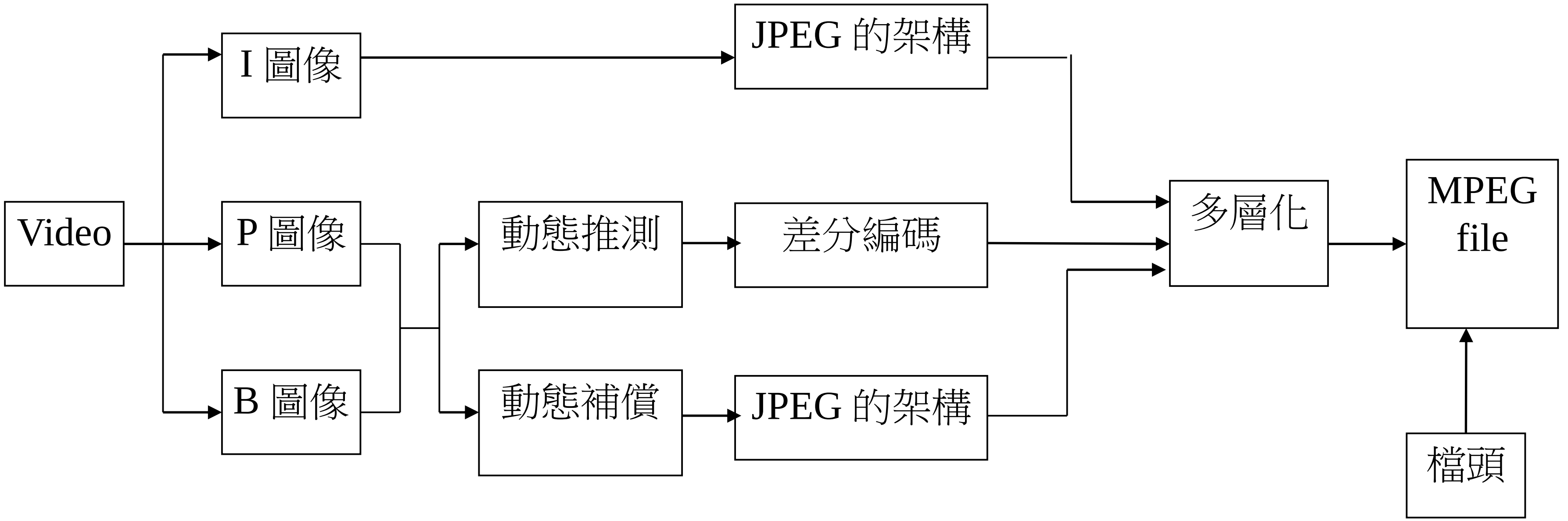
Size: 160×120 Time: 29 sec 一秒 30 個 frames

若不作壓縮： $160 \times 120 \times 29 \times 30 \times 3 = 50112000 = 47.79$ M bytes。

經過 MPEG壓縮： 1140740 = 1.09 M bytes。

只有原來的 2.276%。

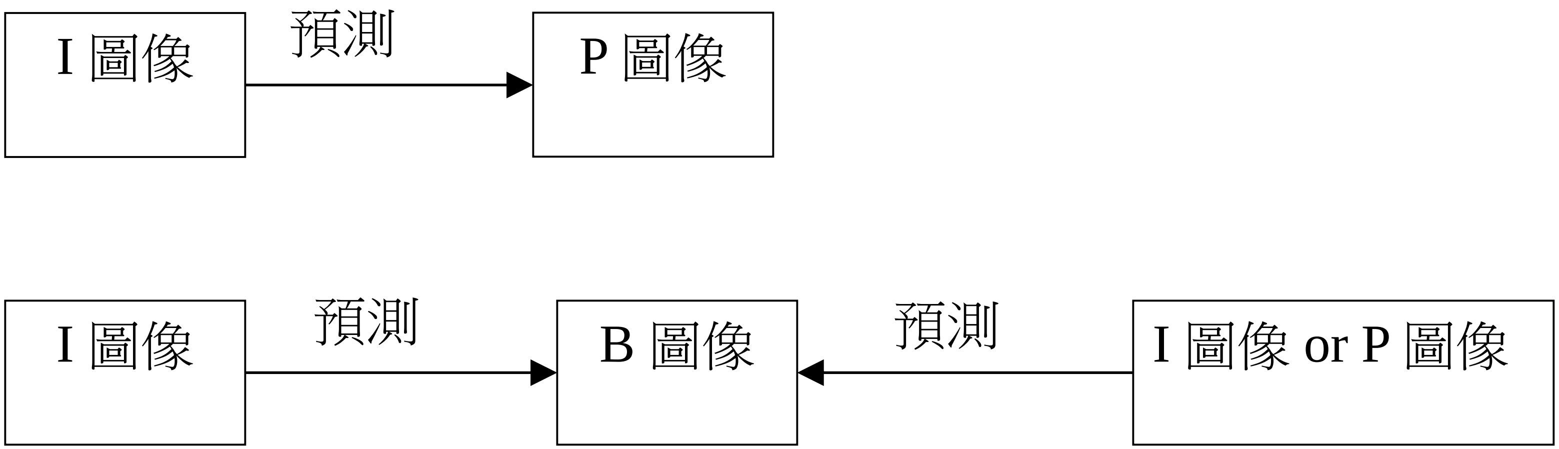
- **Flowchart of MPEG Compression**



I 圖像 (Intra-coded picture): 作為參考的畫格

P 圖像 (Predictive-coded picture): 由之前的畫格來做預測

B 圖像 (Bi-directionally predictive-coded picture): 由之前及之後的畫格來做預測



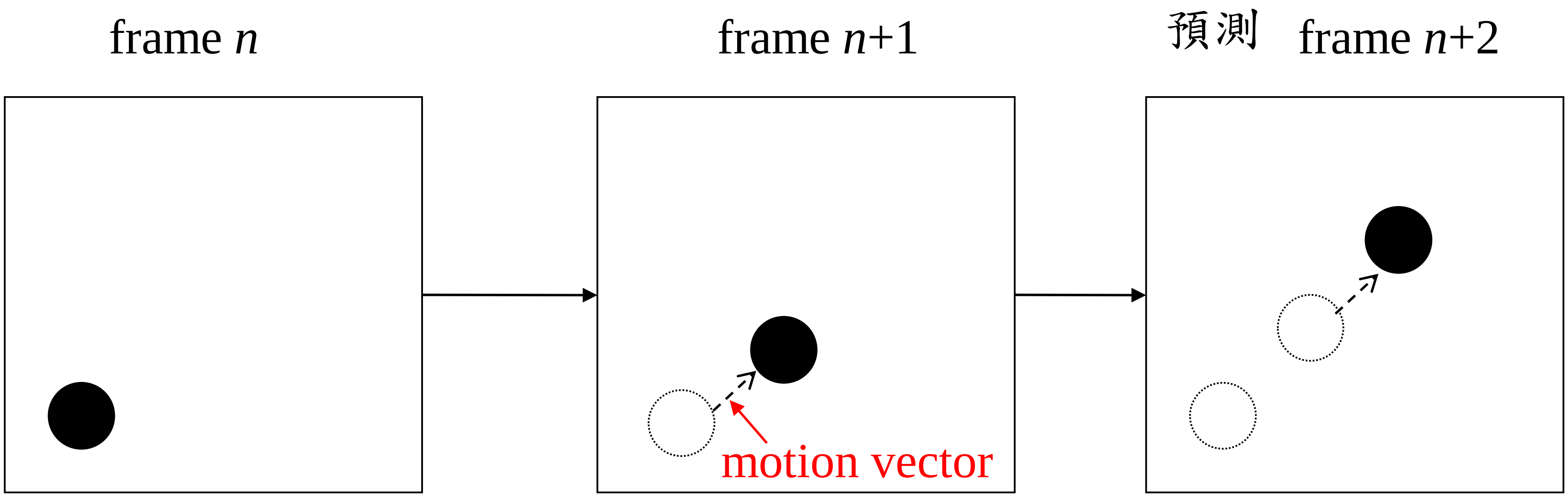
- **動態影像之編碼**

  原理：不同時間，同一個 pixel 之間的相關度通常極高

  只需對有移動的 objects 記錄 “motion vector”

- 動態補償 (Motion Compensation)

  時間相近的影像，彼此間的相關度極高



$F[m, n, t]$: 時間為 t的影像

如何由 $F[m, n, t]$， $F[m, n, t+\Delta]$ 來預測 $F[m, n, t+2\Delta]$?

(1) 移動向量 $V_x(m, n), V_y(m, n)$

(2) 預測 $F[m, n, t+2\Delta]$ ：

$$F_p[m, n, t+2\Delta] = F[m - V_x(m, n), n - V_x(m, n), t+\Delta]$$

(3) 計算「預測誤差」

$$E[m, n, t+2\Delta] = F[m, n, t+2\Delta] - F_p[m, n, t+2\Delta]$$

對預測誤差 $E[m, n, t+2\Delta]$ 做 編碼

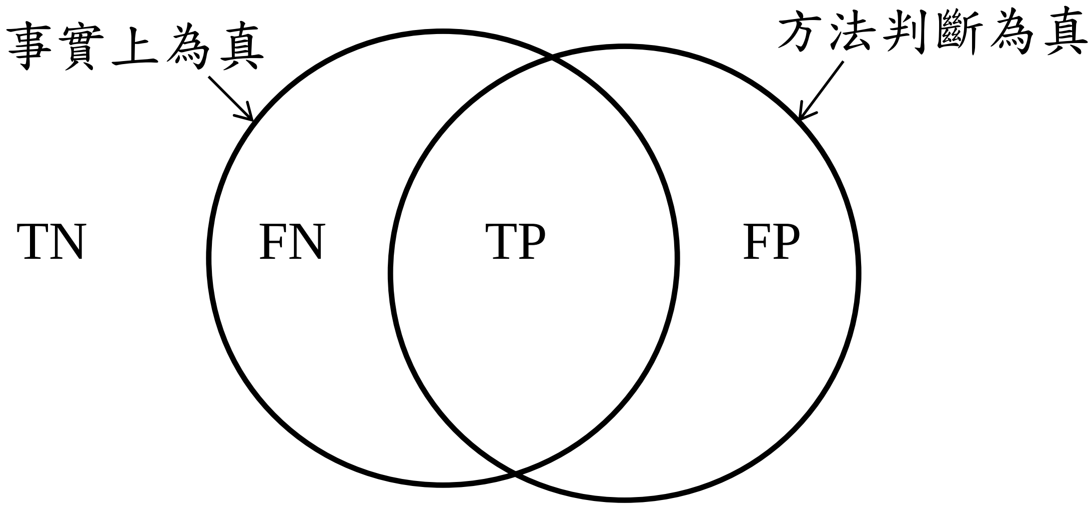
◎ 8-G Data Compression 未來發展的方向

Two important issues:

Q1: How to further improve the compression rate

Q2: How to develop a compression algorithm whose compression rate is acceptable and the buffer size / hardware cost is limited

附錄八：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

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

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

$$specificity = \frac{TN}{TN+FP}$$

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

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
$$\frac{(1+\beta^2)\,precision \times recall}{\beta^2\,precision + recall}$$

附錄九：新的相似度測量工具：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) $$\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|y[m,n]-x[m,n]\right|^2$$

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

(5) L_α-Norm

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

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

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

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

(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, $X_{Max} = 255$

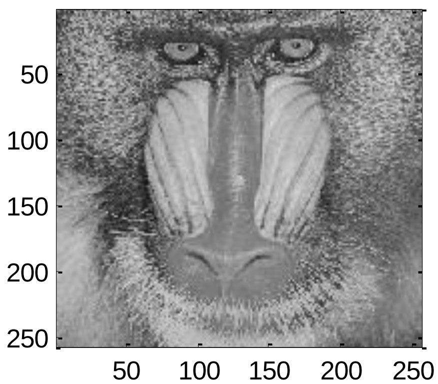
for color image:

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

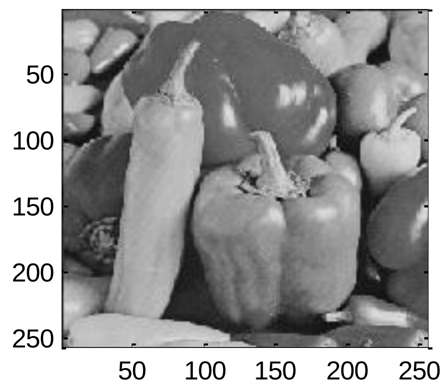
color = R, G, or B

然而，MSE 和 NRMSE 雖然在理論上是合理的，但卻無法反應出實際上兩個信號之間的相似度

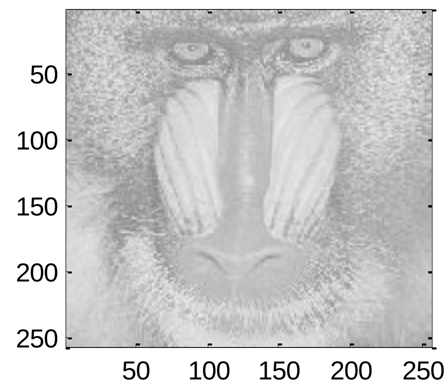
例如：以下這三張圖



圖一



圖二



圖三

圖三 = 圖一 $\times 0.5 + 255.5 \times 0.5$

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

(8) Structural Similarity (SSIM)

有鑑於 MSE 和 PSNR 無法完全反應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

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

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

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

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

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

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

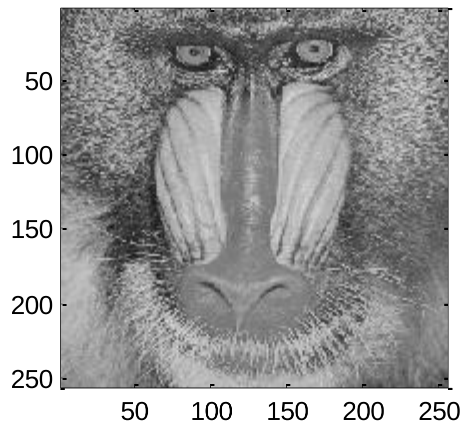
圖一、圖二之間的 SSIM 為 0.1040

圖一、圖三之間的 SSIM 為 0.7720

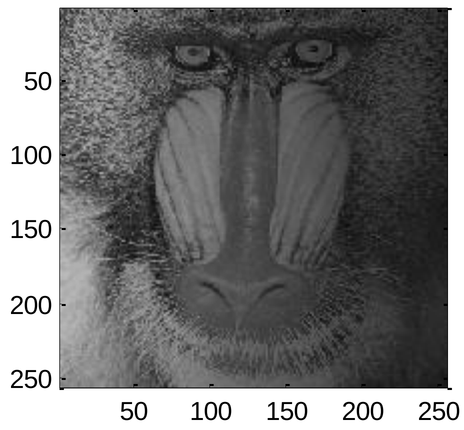
反應出了圖一、圖三之間確實有很高的相似度

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

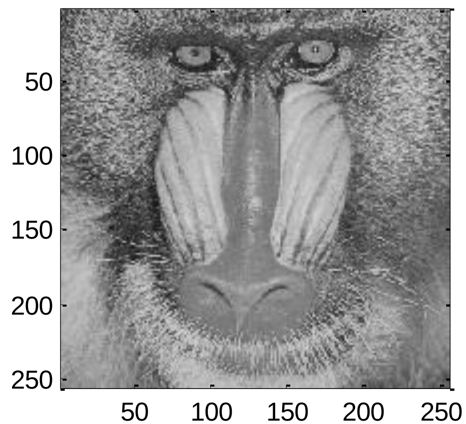


圖五

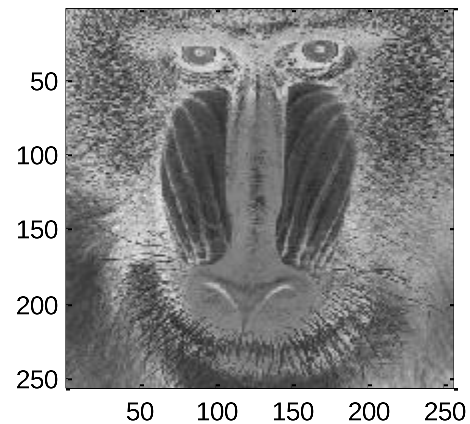
NRMSE = 0.4521 (大於圖一、圖二之間的 NRMSE)

SSIM = 0.6010

底片 the negative of a photo



圖六



圖七

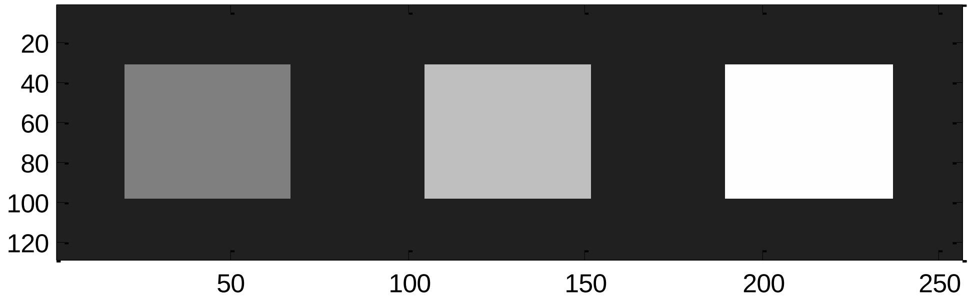
圖七 = 255 − 圖六

NRMSE = 0.5616 (大於圖一、圖二之間的 NRMSE)

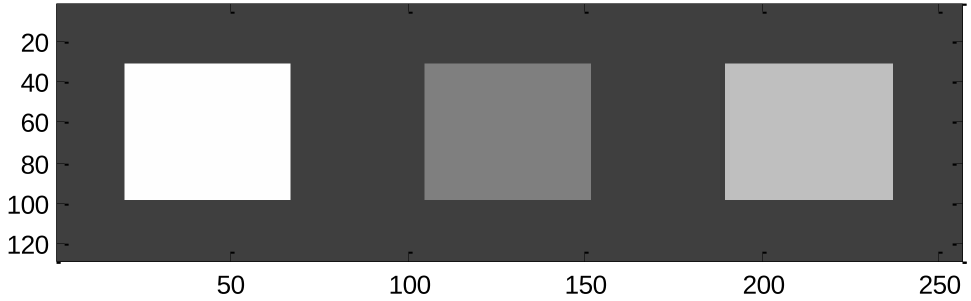
SSIM = -0.8367 (高度負相關)

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 是否真的能反應出兩個信號的相似度？

為什麼？