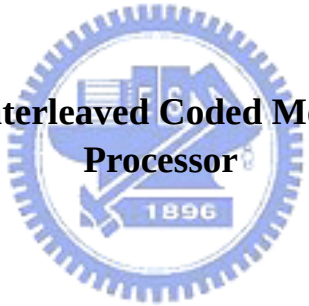
國 立 交 通 大 學

電信工程學系碩士班

碩士論文

使用延遲器之首尾相連式之位元交錯調變碼



On a Tail-Biting Bit-Interleaved Coded Modulation using a Delay Processor

研究生：湯秉熹

指導教授：謝世福 博士

蘇育德 博士

中華民國九十四年三月

使用延遲器之首尾相連式之位元交錯調變碼

On a Tail-Biting Bit-Interleaved Coded Modulation using a Delay Processor

研究生：湯秉熹　　Student：Tang Bin-Ci

指導教授：謝世福　教授　　Advisor：Hsieh She-Fu

蘇育德　教授　　Su Yu Ted

國 立 交 通 大 學

電信工程學系碩士班

碩士論文

A Thesis Submitted to
Department of Communication Engineering
College of Electrical Engineering and Computer Science
National Chiao-Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in

Communication Engineering

March 2005/4/19

Hsinchu, Taiwan, Republic of China

中華民國九十四年三月

使用延遲器之首尾相連式之位元交錯調變碼

研究生：湯秉熹　　指導教授：謝世福 博士
蘇育德 博士

國立交通大學電信工程學系

中文摘要

由於使用延遲器的多層次編碼調變（multilevel coded modulation）需要使用前導位元（pilot bits，通常使用0位元）來幫助解碼，這些多餘的補0位元造成傳輸功率的浪費及淨傳輸率的損失。本論文即針對這項使用延遲器的多層次編碼調變的缺失提出一種改善的結構。我們使用首尾相連（tail-biting）的技術，使之能利用解碼碼字迴授（decoded codeword feedback）在第二次遞迴解碼時將之當成前導位元使用以改善位元錯誤率。這種首尾相連的結構可不需要前導位元，並能利用遞迴式解碼持續改善性能。另外，由於此種編碼調變方式在結構上與位元交錯式調變碼（bit interleaved coded modulation）相似，我們也將其位元錯誤率表現與位元交錯式調變碼之性能做比較。

On a Tail-Biting Bit-Interleaved Coded Modulation Using a Delay Processor

Student: Tang Bin Ci Advisors: Hsieh She-Fu
Su Yu Ted

Institute of Communication Engineering
National Chiao Tung University

Abstract

This thesis proposes a coding scheme based on delay processor interleaved multilevel coded modulation (DPI-MCM). Conventional DPI-MCM requires the insertion of redundant pilot bits (usually "0" bits are used) whence decreases the net data throughput. Our scheme gives a better data throughput for it can do without pilot bits. We use a tail-biting-like structure so that the "tail" part of the decoded output obtained in the initial decoding iteration can be used as the pilot bits in the second iteration in a decision-feedback manner. As DPI-MCM can be regarded as a special case of bit-interleaved coded modulation (BICM), we also compare the performance with compatible interleaving depth.

致謝

本論文得以順利完成，首先要感謝指導老師蘇育德教授，在三年多的研究生活中，不厭其煩地關懷與指導，在此向蘇育德老師致最大的謝忱。並且感謝口試委員呂忠津教授、吳文榕教授、王忠炫教授提供寶貴的意見，彌補了論文的缺失與不足。

三年多，一路停停走走，跌跌撞撞，不斷地考驗著老師的耐心下，在研究遇到瓶頸灰心之際，總能適時地得到老師與實驗室學長學弟妹的關心與鼓勵，特別是李昌明學長和鄭延修學長，在研究繁忙之際還能抽空給予研究上的指導。

最後，要感謝的，就是一直關心我、鼓勵我的家人，僅以此論文獻給所有關心我、愛我的人、愛過我的人，代表我最深的謝意。

目錄

表目錄

圖目錄

第一章

簡介

1.1 編碼調變（Coded Modulation）

傳統的通道編碼，編碼與調變是分開的。在傳送端加入與資訊位元相關的多餘位元（redundant bits），以便接收端能夠利用這種相關性來檢查收到的位元是否有錯，並達成錯誤更正的目的。這種作法會降低資訊傳送的頻譜使用效率。爲了更有效傳遞資訊，溫格伯氏（Ungerboeck）在八十年代初期提出了編碼調變（或稱調變碼）的觀念，其主要的精神是將編碼與調變合併處理，共同設計，並以提高調變碼的最小歐基里德距離（minimum Euclidean distance）爲設計準則。由於調變碼是利用擴張信號星座（signal constellation）大小來取代位元數的擴張，因此大多使用多相鍵移（multiple phase shift keying, MPSK）或二維振幅調變（quadrature amplitude modulation, QAM）的信號。然而，無線通訊環境，例如行動通訊與室內無線網路，本質上具多重路徑傳輸的特性，其通道無法單純地以白色高斯加成性通道（Additive White Gaussian Noise, AWGN）來描述。在多重雷利衰退路徑的通訊環境中，由於MPSK或QAM信號之解調需要準確快速的相位與振幅估計，使得編碼調變的應用無法普及。

本篇論文提出一種新型調變碼，期望能藉著編碼與調變間的延遲器所產生的特殊信號結構與簡單的遞迴解碼，讓解碼器不需要太複雜的運算，亦能在雷利衰退通道（Rayleigh fading channels）下得到相當好的位元錯誤率表現。

本篇論文各章節內容簡述如下：第二章將介紹位元交錯式調變碼[2][3]，以及爲何位元交錯式調變碼採用格雷碼的信號點對應方式。第三章將介紹使用延遲處理器的多層次編碼[5]，以及爲何該編碼方式需要採用集合分割的信號點對應方式。並於第四章提出首尾相連式位元交錯調變碼，以即與前兩章介紹的兩種調變碼在一起做個比較與討論。在第五章中，我們與其他利用位元交錯式調變碼做遞迴解碼的系統做比較，並探討彼此之間的優點與原因，進而得知本篇所提的首尾相連式位元交錯調變碼未來的研究方向。

第二章

位元交錯式調變碼的簡介

2.1 簡介

位元交錯式調變碼（Bit-Interleaved Coded Modulation，BICM），最早是由 Zehavi [2]於 1992 年所提出的一種特殊架構，而在 1998 年，由 Giuseppe Caire, Giorgio Taricco, 與 Ezio Biglieri 三人將之推廣、一般化並命名爲位元交錯式調變碼[3]。由於無線通道，已經不再是白色高斯加成性通道雜訊可以描述的，還要考慮到多重路徑的影響，以及都卜勒效應造成的問題等等。位元交錯式調變碼，一方面可以克服雷利衰退通道造成的接收訊號叢錯（burst error）現象，另一方面並能得到多向度增益（diversity gain），降低通道衰退導致的損失。

在衰退性通道之下，常會遇到訊號的振幅被放大或是縮小（建設性與破壞性干涉）；在慢速衰退（slow fading）之下，遇到嚴重的通道衰退，由於其平均衰退區間長度（mean fade duration）相對的較長，會造成連續一長串的極不可靠符元，導致叢錯的產生。因爲迴旋碼對叢錯的更正能力不高，如何由連串被破壞的訊號中，將原本要傳送的資訊位元給改正、還原回來，一向是通道編碼中重要的課題。位元交錯式調變碼設計的重要基礎概念，就是要將慢速衰退中可能遇到的一連串被破壞的碼字（codewords），利用位元交錯器（bit interleaver）將同一碼字內的碼字位元，打散到同調時間之外，降低一連串碼字被破壞的風險，進而避免叢錯的產生。我們接下來將針對位元交錯式調變碼作基本的介紹。

2.2 符號定義

開始介紹之前，我們先敘述本篇論文會用到的一些符號的定義。在本篇中，每一組碼字符元，均由 M 個位元所組成，而所使用的信號星座集，本篇多半是採用 8 相位鍵移（8-PSK），因此在畫圖或是舉例時，M 通常以 3 來表示。另外我們假設傳送的信號長度是 N，π 代表交錯器的數學函數，μ 爲信號點對應器的對應關係式，ρ 則爲衰退通道的振幅大小。

2.3 位元交錯式調變碼的介紹

之前提到的 Ephraim Zehavi 於 1992 年針對雷利衰退通道所提出的 8-PSK Trellis Codes [1]，就是有鑑於傳統的籬柵式編碼調（Trellis Coded Modulation, TCM），在面臨衰退通道時，所用的交錯器多半是符元對符元（symbol-by-symbol）的。這也就是說：當一組碼字符元（codeword symbol）進入交錯器之後，出來

的還是一組碼字符元，只是在時序上被交錯了；假如被交錯之後的時序，正巧通道情況很糟，該組碼字符元可能就無法正確的幫助解碼了。Zehavi 的想法是：假設一組碼字符元含有 M 個碼字位元，利用 M 個交錯器分別將這 M 個位元交錯到不同的時間點上，或許在這 M 個時間點中，不會全部都遇到很糟糕的通道，而還原回來的碼字，也不至於糟到全部 M 個都錯，可以由此得到多向度增益。有關於傳統的籬柵式調變碼與位元交錯式調變的介紹，將於本節敘述。

傳統的籬柵式編碼調變架構如圖 2.1，在編碼器之後，緊接著就是符元交錯器，再來就是信號點對應器（signal mapper）。首先關於符號的介紹，I是代表輸入編碼器的資訊位元串，C則是編碼完之後的碼字串，I與 C分別可寫作入下所述：

$$I=[i_0^0,i_0^1,i_1^0,i_1^1,i_2^0,i_2^1,\cdots,i_p^0,i_p^1,\cdots]=[I_0,I_1,I_2,\cdots,I_{p,}\cdots]$$
$$C=[c_0^0,c_0^1,c_0^2,c_1^0,c_1^1,c_1^2,\cdots,c_p^0,c_p^1,c_p^2,\cdots]=[C_0,C_1,C_2,\cdots,C_p,\cdots],\ p=0,\cdots,N-1$$

下標 p 代表時間。經過符元交錯器之後的碼字串，我們以 C' 表示，以 π 代表交錯器的函數，原本的時間爲下標 p，交錯過後，時間 p 的符元會被交錯到時間 n，我們以下標 n 代表新的時序。X則是 C' 在信號對應器對應之後，信號空間 Ω 上面的一個信號點，X與 C' 有一對一的關係，也是我們所發送的訊號，我們以 $X=\mu(C')$ 此函示來表示 X與 C' 之間的關係：
$C'=[C'_0,C'_1,C'_2,\cdots,C'_n,\cdots],\ n=0,\cdots,N-1$。其中$C'_n=C_p,n=\pi(p)$
$X=[X_0,X_1,X_2,\cdots,X_n,\cdots],\ n=0,\cdots,N-1$。其中 $X_n=\mu(C'_n)$ 爲信號空間 8 相位鍵移中的一點。

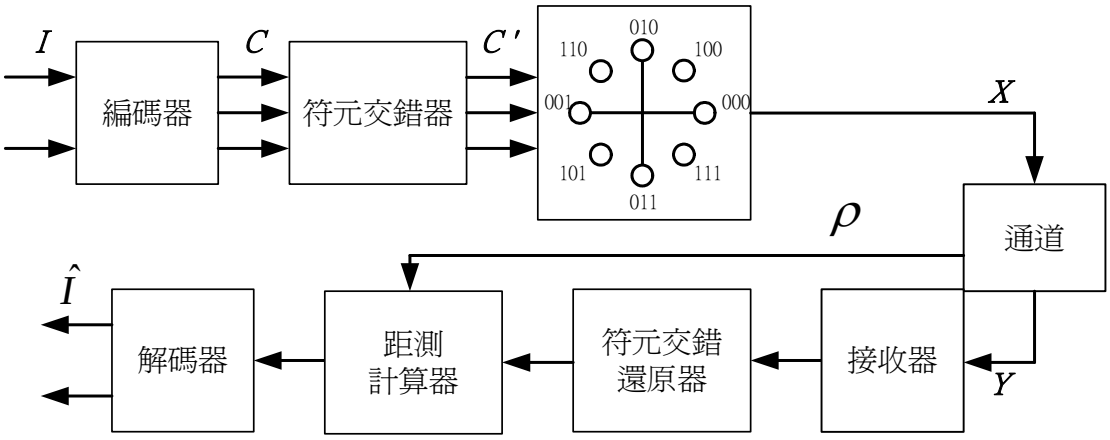


圖 2.1 傳統籬柵式編碼調變的架構

至於通道的狀況，爲了要在雷利衰退通道之下作位元交錯式調變碼與籬柵式調變碼的比較，所以我們在接收端所接到的訊號是寫作以下形式

$$Y=[Y_0,Y_1,Y_2,\cdots,Y_n,\cdots],\ Y_n=\rho_n X_n+w_n,\ n=0,\cdots,N-1$$

其中w_n 爲白色高斯加成性通道雜訊，而 ρ_n 則表示爲在時間n時，通道給予當時訊號X_n的衰減量。交錯器在設計上要超過衰退的同調時間（coherent time），使得這個通道會呈無記憶性（memoryless）的狀態，方便我們作分析（通常是假設交錯器的深度是無窮大）。無記憶性通道的數學式可以寫作如下所示：

$$P(Y \mid I,\rho) = P(Y \mid X,\rho) = \prod_{n=0}^{N-1} P(Y_n \mid X_n,\rho_n), \quad \rho = (\rho_0,\cdots,\rho_n,\cdots,\rho_{N-1}) \qquad (2.1)$$

假設接收端可以得知通道的狀態，也就是 ρ 可以得知，那麼在最大可能性串列估計（Maximum-likelihood sequence estimation）中所需要的距測（metric）便可以寫成

$$m(Y,X;\rho) \equiv \ln P(Y \mid X,\rho) = \sum_{n=0}^{N-1} \ln P(Y_n \mid X_n,\rho_n) = \sum_{n=0}^{N-1} -\left\| Y_n - \rho_n X_n \right\|^2 \qquad (2.2)$$

之後再將每個時間所算出來的距測，交給解碼器利用威特比解碼演算法（Viterbi decoding algorithm）來找出最有可能的資訊位元串 $\hat{I}$ 。

位元交錯式調變碼的架構則如圖 2.2 所示。因爲當時的想法就是想要將同一個符元內的 M 個位元送到不同時間與不同位置來避免 M 個位元都遭到同樣振幅的衰減，所以位元交錯式調變碼的交錯器是採用位元交錯器，將這 M 個位元完美地交錯，於是每個位元都可以看作在 M 個統計獨立的通道上被傳送。其中 C 與 C'的關係與傳統籬柵式編碼調變會有些許不同，如下所示：

$$\begin{aligned} C' &= [c'^{0}_{0}, c'^{1}_{0}, c'^{2}_{0}, c'^{0}_{1}, c'^{1}_{1}, c'^{2}_{1}, c'^{0}_{2}\, c'^{1}_{2}, c'^{2}_{2}, \cdots, c'^{0}_{n}, c'^{1}_{n}, c'^{2}_{n}, \cdots] \\ &= [C'_0, C'_1, C'_2, \cdots, C'_n, \cdots], \quad n = 0,\cdots,N-1 \text{，其中} \\ c'^{j}_{n} &= c^{i}_{p}, \quad \pi : (p,i) \to (n,j), \quad \text{for } j,i = 0,1,\cdots,M-1, \quad n,p = 0,\cdots,N-1 \end{aligned} \qquad (2.3)$$

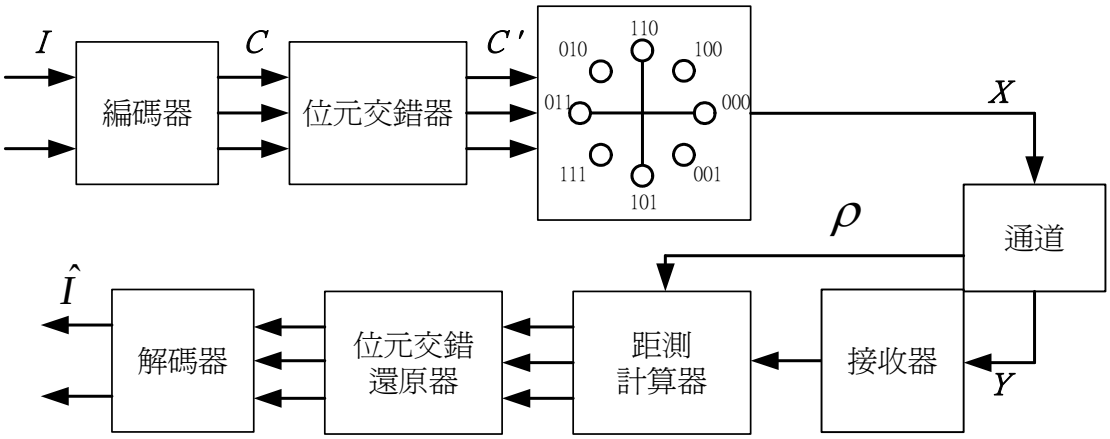


圖 2.2 位元交錯式調變碼的架構

位元交錯式調變碼的距測計算，是將 M 個位元距測（bit metric）算出來之後，加起來起來成爲分支距測（branch metric）送進解碼器用威特比解碼演算法

來找出最有可能的資訊位元串 $\hat{I}$ 。位元交錯式調變碼的特點就在位元距測的計算上面，由於 ***M*** 個位元被分在 ***M*** 個不同的位置上，因此，在每個位置上的位元距測計算都有些許不同。

首先我們先定義以下這些子集（subset），以 8 相位鍵移（***M***=3）爲例，我們將信號空間Ω，根據不同的位元位置（bit position），而切割出以下 3 個子集：

$$S_0^0=\left\{X:X(k)=\sqrt{E_s}\exp(j\frac{k\pi}{4}),k=0,1,2,3\right\},$$
$$S_1^0=\left\{X:X(k)=\sqrt{E_s}\exp(j\frac{k\pi}{4}),k=0,1,4,5\right\},\ \text{與}\ \ S_i^1=\Omega-S_i^0,i=0,1,\cdots,M-1$$
$$S_2^0=\left\{X:X(k)=\sqrt{E_s}\exp(j\frac{k\pi}{4}),k=0,1,3,5\right\},$$

（2.4）

子集之中的所有信號點，其共通關係爲其第 ***i*** 個位元均相同，例如(010)與(111)這兩個信號點在 ***i***=1 的位元均爲 1，因此這兩點都會在 $S_{i=1}^1$ 這個子集合中。另外由條件機率的觀點來看位元距測的計算方法的話，我們要先定義一個函數：

$$l^j(X_n)=b,b\in\{0,1\},\ j=0,1,\cdots,M-1 \qquad (2.5)$$

這表示著第***n***時間傳送的符元***X_n***，它的第***j***個位元爲***b***。因此在已知第***j***位元是***b***，且當時的通道情況爲的 ρ_n 條件機率可寫做如下：

$$P(Y_n\mid l^j(X_n)=b,\rho_n)=\sum_{X_n\in S_j^b}P(Y_n\mid X_n,\rho_n)P(X_n\mid l^j(X_n)=b)$$
$$=\sum_{X_n\in S_j^b}P(Y_n\mid X_n,\rho_n)2^{-(M-1)} \qquad (2.6)$$

假設第***p***時間的碼字是[$c^0{}_p$，$c^1{}_p$，$c^2{}_p$]，經過位元交錯器交錯之後，分別落在n_0，n_1，n_2三個時間的j^0，j^1，j^2位置上，先看$c^0{}_p$這個位元該如何來計算它的位元距測。假設$c^0{}_p$=0，j^0=1，我們發送的符元X_{n0}一定是$S^0{}_1$中 4 個裡面的其中一個，而且其條件機率 $P(X_{n_0}\mid l^{j^0=1}(X_{n_0})=c_p^0)=P(X_{n_0}\mid X_{n_0}=S_1^0)$ 應該是 1/4。

在接收端部分，我們要對收到的Y_{n0}來求它 0 與 1 的位元距測（分別爲收到訊號Y_{n0}在訊號空間上與$S^0{}_1$ 及$S^1{}_1$最近的距離），以幫助我們判斷j^0這個位置的位元到底是 0 或是 1。

$$m_j(Y_n,S_j^b;\rho_n)=-\min_{X\in S_j^b}\left\|Y_n-\rho_nX\right\|^2,\ j=0,1,\cdots,M-1\ ,b=\{0,1\} \qquad (2.7)$$

假設Y_{n0}是落在圖 2.3 中間的斜線區域（該斜線區域內的符元，其中間的位元都是 0），其位元距測計算的結果便對於我們判斷第 1 個位元$c^0{}_p$=0 比較有利。

當*M*個位元的位元距測都算完之後，我們針對第***p***時間有可能的碼字C_p=（$c^0{}_p$，$c^1{}_p$，…，$c^{M-1}{}_p$）將它的位元距測加起來，成爲一個分支距測送入解碼器

來解碼。分支距測的數學式子可寫作如下：

$$m(Y,C_p;\rho)=\sum_{i=0}^{M-1}(1-c_p^i)m_j(Y_n,S_j^0;\rho_n)+c_p^i m_j(Y_n,S_j^1;\rho_n)$$
$$,\ \pi(p,i)=(n,j),\quad i,j=0,1,\cdots,M-1\quad n,p=0,1,\cdots,N-1 \qquad (2.8)$$

我們將分支距測累加值 $m(Y,C;\rho)=\sum_{p=0}^{N-1}m(Y,C_p;\rho)$ 送進威特比解碼演算法，由最大的分支距測累加值，來決定其相對應的碼字串 $\hat{C}$ 與資訊位元串 $\hat{I}$ ，便是最有可能的結果。

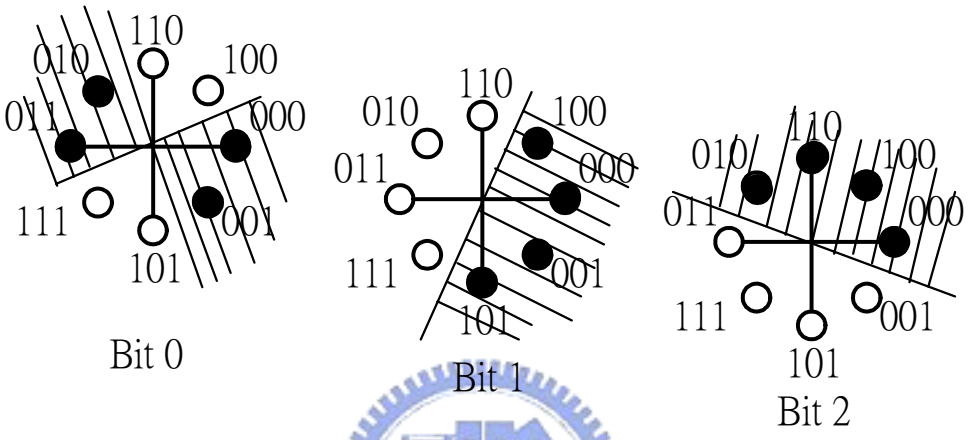


圖 2.3 格雷碼對應方式下不同位置的子集分割方式

2.4 格雷碼的介紹

格雷碼（Gray code），是位元交錯式調變碼常採用的信號點對應方式，設計原則敘述如下：

假設Ω爲信號點空間，信號點的數目爲 2^M，亦即用M個位元就能描述信號空間中的一點，且信號點間彼此最短的距離爲$d_{\min}$；而信號空間中的子集合 S_i^b ，$b\in\{0,1\}$，$i=0,\cdots,M-1$，其定義如同（2.4）式所描述。若信號點空間Ω中的任何一點 $x\in S_i^b$ 最多只有一個信號點 z 距離 x 爲$d_{\min}$，其中 $z\in S_i^{\bar{b}}$ ，爲 S_i^b 的互補子集合中的一點，此時的信號對應方式，稱做格雷碼對應方式。

。

讓我們試著由位元交錯式調變碼的位元距測計算方法，來瞭解爲何格雷碼是位元交錯式調變碼常用的信號點對應方式。由（2.7）式我們知道位元距測的計算，是由兩個子集中分別找出最接近接收訊號$\boldsymbol{Y_n}$的信號點 $\rho_n\boldsymbol{X}$，因此假若$\boldsymbol{Y_n}$旁的 z 最多只有一個（ $z\in S_i^{\bar{b}}$ ），那麼對於第i位元的判斷將有利於b。我們以集合分割（set partitioning）與格雷碼的第 2 個位元做例子，如圖 2.4 所示，紅色線段的長短分別代表位元距測的大小。

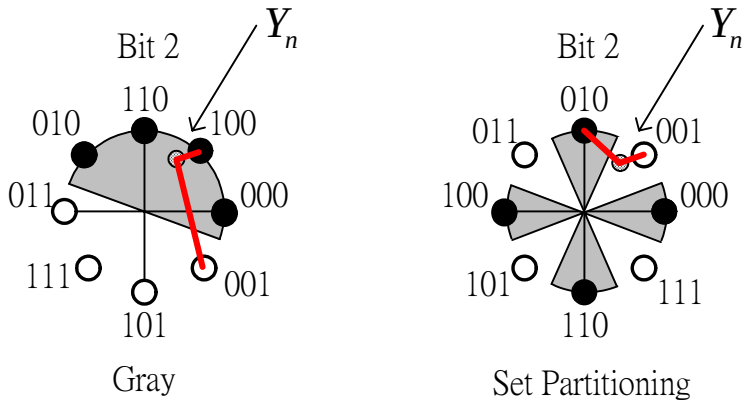


圖 2.4 格雷碼與集合分割在最右側位元的位元距測計算

範例 2.1 假設爲 8 相位鍵移，如圖 2.4，接收到的$\boldsymbol{Y_n}$信號座標爲(0.4，0.7)，我們分別計算其b=0 與b=1 的位元距測。

$$m_j(Y_n, S_j^b; \rho_n) = -\min_{X \in S_j^b} \|Y_n - \rho_n X\|^2 , j = 2 \ , b = \{0,1\} \Rightarrow$$

格雷碼對應：$\left\{ \begin{array}{l} m_2(Y_n, S_2^0; 1) = -\min\limits_{X \in S_2^0} \|Y_n - X\|^2 = -0.0944 \\ m_2(Y_n, S_2^1; 1) = -\min\limits_{X \in S_2^1} \|Y_n - X\|^2 = -2.16855 \end{array} \right\} \Rightarrow$ 位元距測差異: 2.07415

集合分割對應：$\left\{ \begin{array}{l} m_2(Y_n, S_2^1; 1) = -\min\limits_{X \in S_2^1} \|Y_n - X\|^2 = -0.0944 \\ m_2(Y_n, S_2^0; 1) = -\min\limits_{X \in S_2^0} \|Y_n - X\|^2 = -0.25 \end{array} \right\} \Rightarrow$ 位元距測差異: 0.1556

可以看到，若是子集內的信號點太過分散，以上圖右側的集合分割爲例，此時接收的信號$\boldsymbol{Y_n}$，其b=0 與b=1 的位元距測差異不大，對於威特比解碼器中，每一條碼字路徑（codeword path）的殘存距測（survival metric）彼此之間差異不大的結果，將會導致訊號叢錯發生的可能性提高。而格雷碼對應方式會有子集內信號點較集中的特性，因此在使用位元交錯式調變碼時，我們多半使用格雷碼對應。

2.5 位元交錯式調變碼的討論

圖 2.5 與 2.6 爲位元交錯式調變碼的模擬結果，雷利衰退通道爲傑克衰退通道模型所產生，又模擬的假設條件爲表 2.1 所示，且假設信號相位在接收端完全已知。

調變（modulation）	8PSK
標準化都卜勒頻率 fc_T （normalized Doppler frequency）	0.01
State 個數	8
碼字位元/封包 （codeword bits per package）	2001
位元方塊交錯器交錯深度	75

表 2.1 BICM 的模擬參數

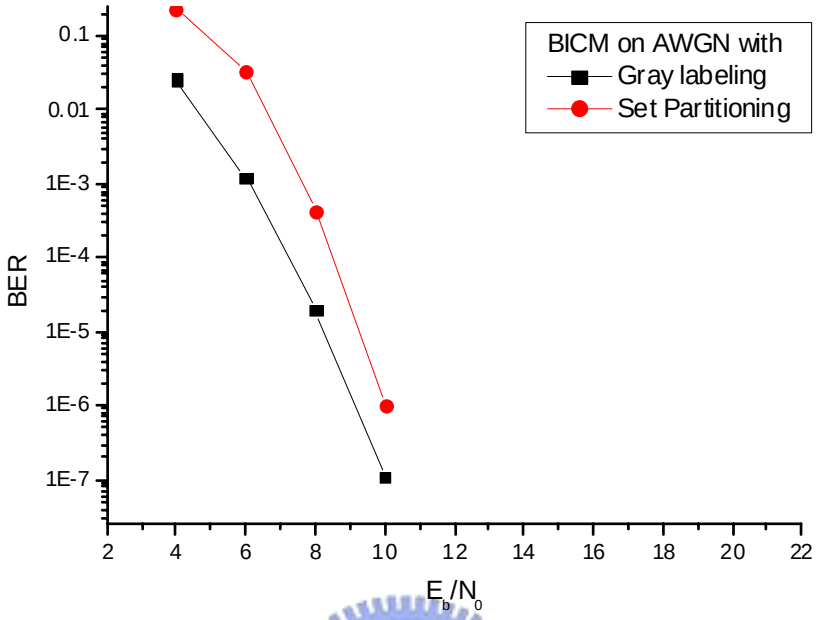


圖 2.5 在白色高斯加成性通道下，位元交錯式調變碼使用不同的信號點對應方式（格雷碼與集合分割）

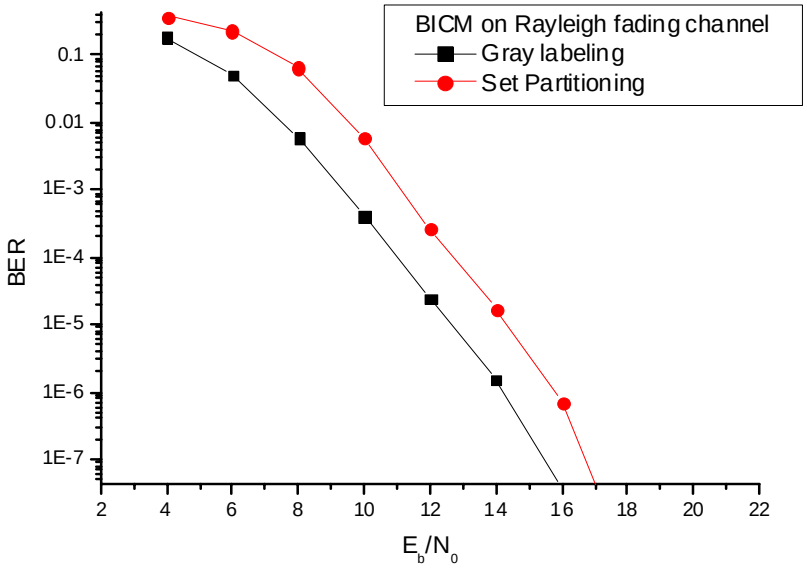


圖 2.6 在雷利衰退通道下，位元交錯式調變碼使用不同的信號點對應方式（格雷碼與集合分割）

由圖 2.5、2.6 與範例 2.1，我們可以瞭解到爲何位元交錯式調變碼多半採用格雷式信號對應。

但是格雷碼信號點對應方式在遞迴式解碼（iterative decoding）的機制下，就無法得到太多的遞迴增益（iterative gain）了。一般來說，隨著遞迴次數增加，同一符元下的其他位元的可信度亦會增加，但是對於格雷碼來說的話並無法將之利用來拉高信號點之間的最小距離。關於這點的討論，我們會在第三章介紹到集合分割時探討。

第三章

多層次編碼（Multilevel Coding）

3.1 簡介

多層次編碼是編碼調變的設計中經常被使用的一種技術。傳統的多層次編碼，其編碼架構如圖 3.1 所示。假設信號集有 2^m個信號點，我們可以將其分割成 m層的架構。在傳統的多層次編碼設計中，我們使用m個二元碼來對這m層的分割結構加以編碼。

在本章中，我們將介紹一種由 Hellstern 於 1993 年提出的多層次編碼，其基本上也是將信號集分割成 m 層的架構，但如圖 3.2 所示，只用一個二元迴旋碼 C（binary convolutional code）來編碼。由溫格伯氏所設計出的籬柵編碼調變，其編碼便是由一個二元迴旋碼和一個信號點對應器所構成。而 Hellstern 的編碼設計，便是在二元迴旋碼和信號點對應器中加入一個延遲處理器（delay processor）。在 Hellstern 的設計中，由於延遲處理器的使用，設計出來的籬柵碼，其限制長度（constraint length）會變得更長。藉由適當地設計延遲處理器，我們可以得到更大的自由距離（free distance）。使用這種方式所設計出來的籬柵碼，可以使用其二元迴旋碼 C 的籬柵圖以及之前已經解碼出來的資訊迴授來加以解碼，其解碼方法不會太複雜，並且該編碼系統也可以達到良好的除錯性能。其缺點則是解碼的延遲長度變得更長。有關傳統多層次編碼、傳統籬柵編碼，與 Hellstern 提出的多層次編碼，我們將於圖 3.3 一併比較。

由於 Hellstern 提出的多層次編碼中之延遲處理器可以視作一種特殊的交錯器，因此 Hellstern 編碼可看成位元交錯式調變碼的一個特例。本章的第三節會將前者與第二章提到的位元交錯式調變碼做比較，討論其於白色高斯加成性通道與雷利衰退通道之下的除錯能力。

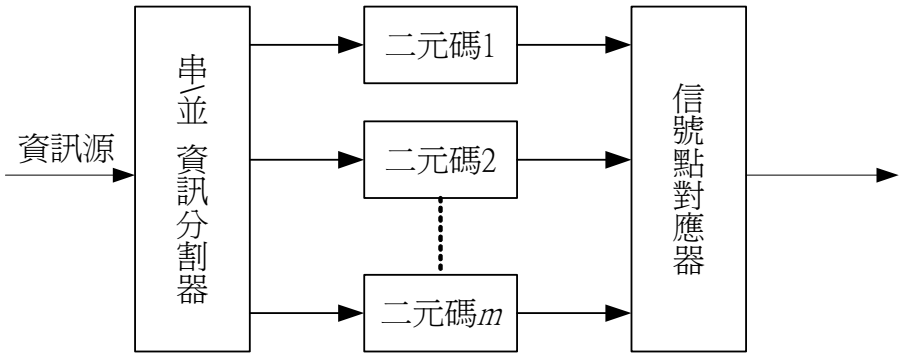


圖 3.1 傳統的多層次編碼其編碼端示意圖

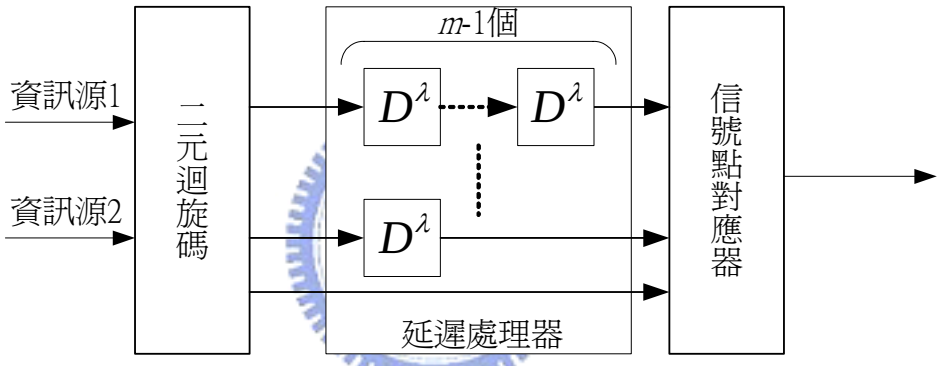


圖 3.2 Hellstern 的多層次編碼架構圖

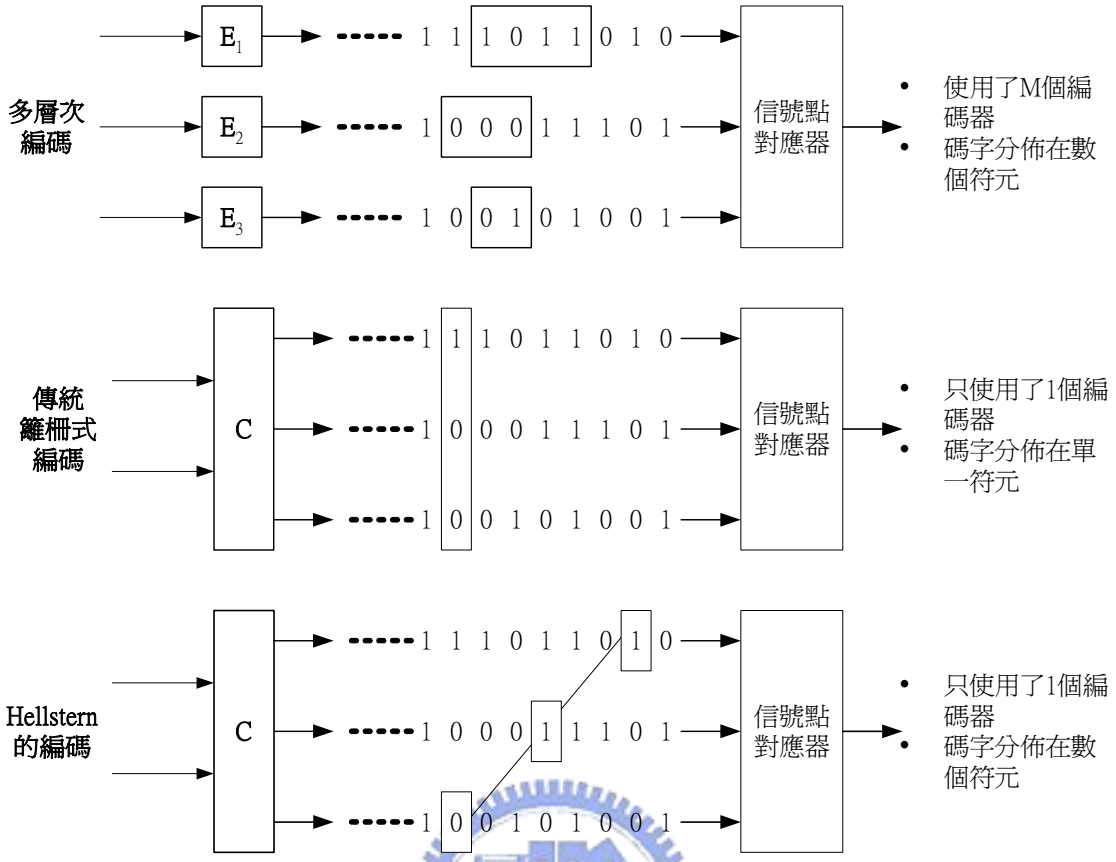


圖 3.3 在不同的編碼技術下，碼字的分佈圖。其中同一組方框代表同一組碼字。

3.2 由一個二元碼和迴旋處理器所架構而成的多層次編碼

在 1993 年，Hellstern [5]提出了一種架構，與傳統的籬柵編碼不同處在於，其在傳統的籬柵編碼調變裡面的編碼端做了修改，如圖 3.4 所示，於二元迴旋碼與信號點對應器之間，加上了 m 層次的的延遲處理器。

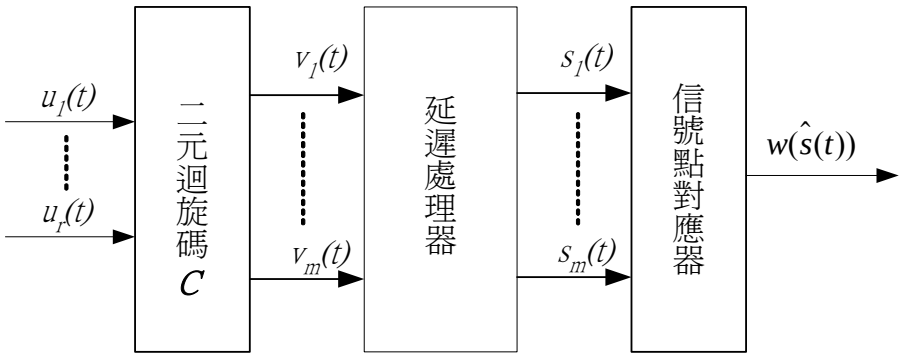


圖 3.4 Hellestern 提出的編碼架構

二元迴旋碼編碼器 C（碼率爲 r/m）的輸入與輸出分別表示爲 $\bar{u} = \{\cdots, \hat{u}(0), \hat{u}(1), \cdots\}$ 與 $\bar{v} = \{\cdots, \hat{v}(0), \hat{v}(1), \cdots\}$，其中 $\hat{u}(t) = (u_1(t), \cdots, u_r(t))$ 是個含有 r

個位元的資訊符元，而 $\hat{v}(t)=(v_1(t),\cdots,v_m(t))$ 是含有 m 個位元的碼字符元。碼字串 $\bar{v}$ 緊接著輸入了延遲處理器，其產生出來的碼字串 $\bar{s}$ 可寫作 $\bar{s}=\{\cdots,\hat{s}(0),\hat{s}(1),\cdots\}$，其中 $\hat{s}(t)=(s_1(t),\cdots,s_r(t))$，與 $\hat{v}(t)$ 一樣是個含有 m 個位元的碼字符元。$v_j(t)$ 與 $s_j(t)$ 的關係可表示如下：

$$s_j(t)=v_j(t-\tau_{mj}),\quad \tau_{mj}=(m-j)\lambda,\quad \text{for } 1\le j\le m \qquad (3.1)$$

延遲過的碼字串接著輸入了信號點對應器，其輸出的符元串可表示作 $\bar{z}=\{\cdots,z(0),z(1),\cdots\}$，其中 $z(t)=w(\hat{s}(t))\in\Omega$，在此處的信號空間 Ω 包含了 2^m 個信號點。Ω 可以是如MPSK、MQAM（$M=2^m$）的那種信號星座集，或是m個位元的二元集合，也就是{0，1}^{m}，不過在本篇論文中，我們只討論第一種的信號星座結構。

3.2.1 集合分割（Set Partitioning）

在討論編碼架構之前，我們先說明一下集合分割的概念。假設信號空間 Ω 是由 2^m 個信號點 $\{z_1,z_2,\ldots,z_2{}^m\}$ 構成的，若我們將信號空間切割成彼此不重疊（nonoverlap）的子集合（subsets），且這些子集合連集起來剛好就是信號空間 Ω，這個切割信號空間的舉動就成了一個分割（partition），我們在此將第一次分割所切出來的子集合稱作 Ω_1。子集合 Ω_1 可以再更進一步分割下去，直到最後的子集合 Ω_q 裡面只有一個信號點。而這一連串的分割，我們可以稱它作分割鏈（partition chain），信號空間 Ω 中的任何一個信號點，均可由此分割鏈 $\Omega_0/\Omega_1/\Omega_2/\cdots/\Omega_q$ 分割而得。

假設 Ω_{j-1}/Ω_j 的分割可以將子集合 Ω_{j-1} 割出 2^{hj} 個子集合 Ω_j，我們則可以用 h_j 個位元來區別這幾個不同的子集合 Ω_j。假如 $h_1+h_2+\cdots+h_q=m$，那我們就可以用 m 個位元來描述信號空間 Ω 中的任何一點。在Hellstern提出的多層次編碼中，$q=m$，而且對於所有的 j 來說，$h_j=1$。

現在，我們有兩個不同的 m 位元的符元 $\hat{s}$ 與 $\hat{s}'$ 要比較其信號空間上的距離，$\hat{s}=(s_1,s_2,\cdots,s_m)$，$\hat{s}'=(s'_1,s'_2,\cdots,s'_m)$，所有的位元 i（$i<j$）均有 $s_i=s_i'$ 的特性，根據不同的相異位元 j，其子集合在信號空間中的最小距離也會跟著不同：

$$\Delta_j(0,1)=\begin{cases}\min\limits_{s_j=0,s'_j=1}\{\Delta(w(\hat{s}),w(\hat{s}')):w(\hat{s}),w(\hat{s}')\in\Omega\}, & \text{if } j=1\\ \min\limits_{s_j=0,s'_j=1}\{\Delta(w(\hat{s}),w(\hat{s}')):w(\hat{s}),w(\hat{s}')\in\Omega,\ \text{and} s_i=s'_i\ \text{for } 1\le i<j\}, & \text{if } 1<j\le q\end{cases} \qquad (3.2)$$

$\Delta_j(0,1)$ 代表所有可能的 $w(\hat{s}),w(\hat{s}')$ 中，距離最短的那一個（可以簡化作 Δ_j 來表示），其中 $w(\hat{s})$ 與 $w(\hat{s}')$ 均在相同的 Ω_{j-1} 裡，而 $w(\hat{s})$ 落在被分割後第j個位元為 0 的子集合 Ω_j 中，$w(\hat{s}')$ 落在被分割後第j個位元為 1 的子集合 Ω_j 中。

由（3.2）式我們可以瞭解到這 m 層次的分割架構有著如下的關係：

$$\Delta_1 \leqq \Delta_2 \leqq \cdots \leqq \Delta_m \qquad (3.3)$$

這種最小自由距離（minimum free distance）會隨著層次（level）的增加而變大的特性，這對往後的分析會有所幫助。關於以上的敘述，可以參考範例 3.1 來幫助釐清。

範例 3.1 假設信號空間Ω是相位鍵移的信號集。若我們對其作 3 層次的分割（亦即$h_1=h_2=h_1=1$），則我們可以看到如圖 3.5 所示，其$\Delta_1=0.586$，$\Delta_2=2$，$\Delta_3=4$。

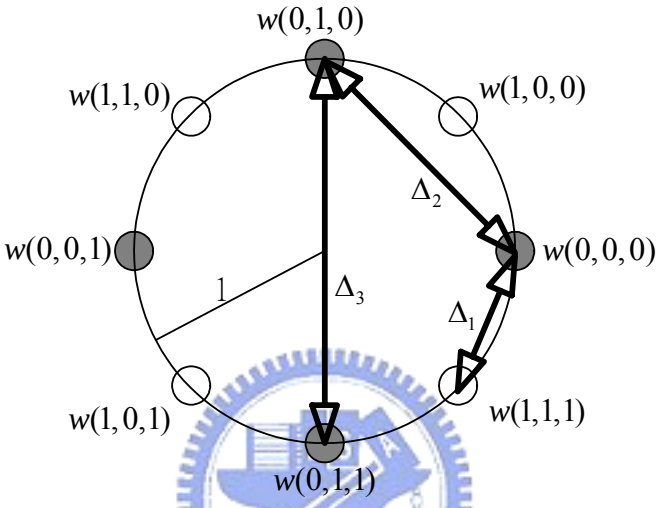


圖 3.5 8 相位鍵移信號集

3.2.2 延遲長度固定的延遲處理器

Hellstern 的編碼架構，在相鄰的層次間，均差異著一個固定的延遲長度 λ。由圖 3.3 與（3.1）式，我們把重點放在延遲處理器的前後，試以圖 3.6 來表示$\bar{v}$與$\bar{s}$的關係。

	$\hat{s}(t)$	$\hat{s}(t+\lambda)$	$\hat{s}(t-(m-j)\lambda)$	$\hat{s}(t+(m-1)\lambda)$
s_1	$v_1(t-(m-1)\lambda)$	$v_1(t-(m-2)\lambda)$	$v_1(t-(j-1)\lambda)$	$v_1(t)$
s_j	$v_j(t-(m-j)\lambda)$	$v_j(t-(m-j-1)\lambda)$	$v_j(t)$	$v_j(t+(j-1)\lambda)$
s_{m-1}	$v_{m-1}(t-\lambda)$	$v_{m-1}(t)$	$v_{m-1}(t+(m-j-1)\lambda)$	$v_{m-1}(t+(m-2)\lambda)$
s_m	$v_m(t)$	$v_m(t+\lambda)$	$v_m(t+(m-j)\lambda)$	$v_m(t+(m-1)\lambda)$

圖 3.6 延遲處理器的$\bar{v}$與$\bar{s}$關係圖

假設分別有兩組碼字串要比較，分別爲 $\bar{v}=\{\cdots,\hat{v}(0),\hat{v}(1),\cdots\}$ 與

$\bar{v}' = \{\cdots, \hat{v}'(0), \hat{v}'(1), \cdots\}$，而其分別對應的輸出符元串分別爲 $\bar{z}$ 與 $\bar{z}'$。假設 $\hat{v}(t)$ 與 $\hat{v}'(t)$ 在 $t<0$ 時都相同，於 $t=0$ 該點出現分歧 $\hat{v}(t=0) \neq \hat{v}'(t=0)$。如此的話，在 0 ≦$t<$ λ，只要有 $i<j$ 這個關係存在，則 $t+\tau_{mj}-\tau_{mi} = t-(j-i)\lambda < 0$。
因此：

$$\begin{aligned} s_i(t+\tau_{mj}) &= v_i(t+\tau_{mj}-\tau_{mi}) \\ &= v_i'(t+\tau_{mj}-\tau_{mi}) \\ &= s_i'(t+\tau_{mj}) \quad \text{for } i<j \end{aligned} \tag{3.4}$$

上面這則關係式（3.4）代表 $\bar{s}$ 與 $\bar{s}'$ 這兩個符元串在灰色框框的左方（代表產生分歧的時間t=0 之前）都是相同的。因此我們可以得知，在灰色框框之上的位元，都是相同的；也就是代表 $z(t+\tau_{mj})$ 與 $z'(t+\tau_{mj})$ 會座落在相同的子集合 $\Omega_{j\text{-}1}$。由（3.2）與（3.3）式可知

$$\begin{aligned} \Delta(z(t+\tau_{mj}), z'(t+\tau_{mj})) &\geq (s_j(t+\tau_{mj}) \oplus s_j'(t+\tau_{mj}))\Delta_j \\ &= (v_j(t) \oplus v_j'(t))\Delta_j \end{aligned} \tag{3.5}$$

因此，符元串 $\bar{z}$ 與 $\bar{z}'$ 的兩兩相對距離（pairwise distance），其下限可以寫做

$$\begin{aligned} \Delta_{LB}(\bar{v}, \bar{v}', \lambda) &= \sum_{t=0}^{m\lambda-1} \Delta(z(t), z'(t)) \\ &= \sum_{j=1}^{m} \sum_{t=0}^{\lambda-1} \Delta(z(t+\tau_{mj}), z'(t+\tau_{mj})) \\ &\geq \sum_{t=0}^{\lambda-1} \sum_{j=1}^{m} (v_j(t) \oplus v_j'(t))\Delta_j \end{aligned} \tag{3.6}$$

我們可以由下面的範例 3.2 來瞭解兩兩相對距離的計算方式。

範例 3.2 若假設 8 相位鍵移，則m＝3，且 $\hat{v}(t)$ 與 $\hat{v}'(t)$ 在$t<0$ 時都相同（均爲 $(000)^{\mathrm{T}}$），於 $t=0$ 該點出現分歧，$\hat{v}(t=0) \neq \hat{v}'(t=0)$，例如 $\hat{v}(0) = (100)^{\mathrm{T}}$，$\hat{v}(1) = (110)^{\mathrm{T}}$，$\hat{v}(2) = (111)^{\mathrm{T}}$，而 $\hat{v}'(0) = \hat{v}'(1) = \hat{v}'(2) = (000)^{\mathrm{T}}$。假設延遲時間 $\lambda = 3$，則碼字串 $\bar{v}$ 與 $\bar{v}'$ 將被延遲處理器轉換成 $\bar{s}$ 與 $\bar{s}'$，如圖 3.7 所示。

若 8 相位鍵移的信號集採如圖 3.5 那種集合分割的信號對應方式，我們可以得到輸出符元串 $\bar{z}$ 與 $\bar{z}'$ 之間的距離平方。其中 $\Delta_1 = 0.586$，$\Delta_2 = 2$，$\Delta_3 = 4$，因此我們可以得知 $\Delta(\bar{z}, \bar{z}')$ ＝0.586x3+2x2+4x1＝9.758。

$\Delta(z(0), z'(0)) = 0$	$\Delta(z(1), z'(1)) = 0$	$\Delta(z(2), z'(2)) = \Delta_3$
$\Delta(z(3), z'(3)) = 0$	$\Delta(z(4), z'(4)) = \Delta_2$	$\Delta(z(5), z'(5)) = \Delta_2$
$\Delta(z(6), z'(6)) = \Delta_1$	$\Delta(z(7), z'(7)) = \Delta_1$	$\Delta(z(8), z'(8)) = \Delta_1$

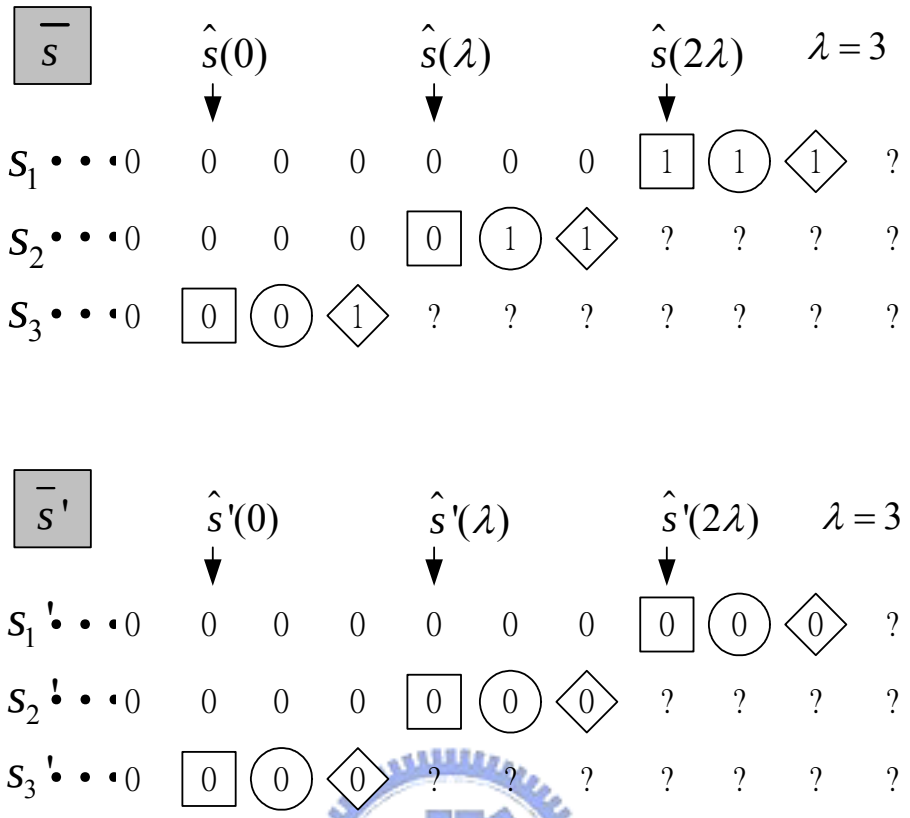


圖 3.7 範例 3.2 中 $\bar{s}$ 與 $\bar{s}'$ 的關係圖

觀察圖 3.7 與上述範例，我們可以知道最小自由距離的增加，主要是來自最下層的相異位元所帶來的貢獻，這也正是 Hellstern 的多層次編碼架構與集合分割此種信號對應方式巧妙的結合，才會有這種結果。

若我們假設限制長度爲 λ，則最小自由距離的下限可以由（3.6）式推得定理 3.1

定理 3.1 當 $\hat{u}(t) = \hat{u}'(t),\quad \text{for } t < 0$ 且 $\hat{u}(0) \neq \hat{u}'(0)$，則其最小自由距離的下限可以寫做

$$\Delta_{LB}(\lambda) = \min_{\substack{v,v' \in C \\ \hat{v}(0) \neq \hat{v}'(0)}} \sum_{t=0}^{\lambda-1} \sum_{j=1}^{m} (v_j(t) \oplus v_j'(t)) \Delta_j \qquad (3.7)$$

由上式可以看到 $\Delta_{LB}(\lambda)$ 是一個會隨著 λ 增加而單調增加的函式，但是若符元串 $\bar{v}$ 與 $\bar{v}'$ 最短可以重新合併起來的時間長度是 $\lambda_{\min}$；也就是代表當限制長度 λ 超過 $\lambda_{\min}$ 之後，不管 λ 再怎麼增加，$\Delta_{LB}(\lambda)$ 也不會隨著增加了。此時的值 $\Delta_{LB}(\lambda_{\min})$，也就是(3.7)式的上限。在本篇的模擬中，我們所用的 λ 均超過 $\lambda_{\min}$。

關於接收端的解碼方式。因爲有考慮到多層次編碼與位元交錯式調變碼，在延遲處理器與交錯器上面有某種程度的類似情形，因此我們參考了王佳盈博士[1]，針對 Hellstern 提出的多層次編碼而提出的次佳解碼方法。此解碼方式的複雜度並不會太高，而且可以利用延遲的特性，來讓距測計算時有可信賴的上層位

元，進而拉大信號點間的最小距離。其解碼方法的主要精神在於利用先前解碼而得的碼字符元，來幫助目前位元距測的計算。且由於在同一符元中，較上層的碼字位元具有較高的可信度，也就如同（3.2）式與（3.3）式，利用集合分割的特性，來拉大位元距測。

我們以範例 3.2 來說明。首先我們先假設 $y(t)$是接收端收到加入了通道所給的雜訊的信號，其數學表示式如下

$$y(t) = \rho(t)z(t) + w(t) \qquad (3.8)$$

其中 $\rho(t)$ 代表通道的衰減量，$w(t)$代表白色高斯雜訊。

第一步 我們先針對 $v_j(t) = 0$ 或 1, $j = 1,2,\cdots,m$，來計算第 j 位元的位元距測 $M_{v_j(t)=0}$ 與 $M_{v_j(t)=1}$ 。在計算位元距測 $M_{v_j(t)}$ 時，會用到 $y(t+\tau_{mj})$ 與先前的解碼結果 $v_i(t-\tau_{ji}),\ 1 \le i < j$ 。 $M_{v_j(t)}$ 代表著接收訊號 $y(t+\tau_{mj})$ 與有可能的信號點 $w(\hat{s}) = w(s_1, s_2, \cdots, s_m)$ 之間的距離，其中的 $\hat{s}$ 滿足以下的特性：

$$\begin{cases} s_i = v_i(t-\tau_{ji}) & \text{for } i < j \\ s_i = v_j(t) & \text{for } i = j \\ s_i \in \{0,1\} & \text{for } i > j \end{cases} \qquad (3.9)$$

第一步的過程可以參考圖 3.8 所示

第二步 計算分支 $\hat{v}(t) = (v_1(t), v_2(t), \cdots, v_m(t))$ 的分支距測 $M_{\hat{v}(t)}$ 。分支距測 $M_{\hat{v}(t)}$ 的計算方法是累加所有可能的 $M_{v_i(t)}$, $i = 1,\cdots,m$ ，也就是

$$M_{\hat{v}(t)=(0,0,0)} = M_{v_1(t)=0} + M_{v_2(t)=0} + M_{v_3(t)=0}$$

$$\vdots$$

$$M_{\hat{v}(t)=(1,1,1)} = M_{v_1(t)=1} + M_{v_2(t)=1} + M_{v_3(t)=1}$$

第三步 將 $M_{\hat{v}(t)}$ 送入威特比解碼器來解迴旋碼 C，因爲其限制長度爲 λ ，所以我們可以解得 $\hat{u}(t-\lambda+1)$ 與 $\hat{v}(t-\lambda+1)$ ，並再將 $\hat{v}(t-\lambda+1)$ 交給下一次使用。

因此我們可以知道，從收到 $y(t=0)$ 直到解出 $\hat{u}(t=0)$ 爲止，總共需要延遲 $m\lambda$ -1 個時間點，這也是使用延遲處理器的多層次編碼會遇到的最大缺點。

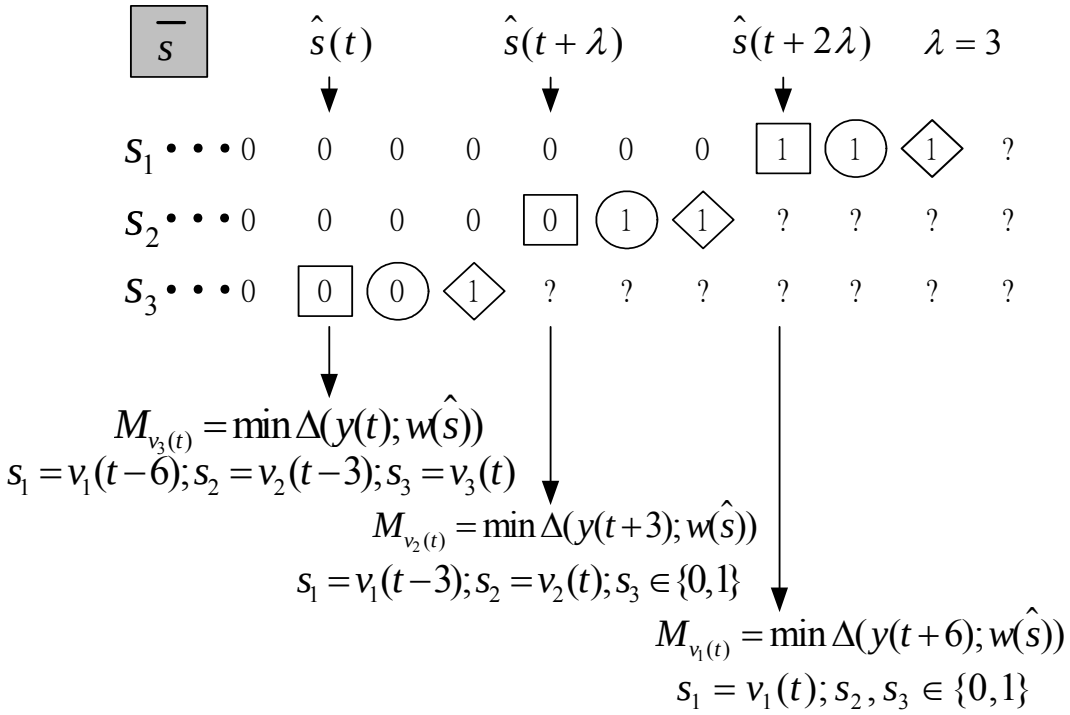


圖 3.8 解碼步驟示意圖

3.3 多層次編碼與位元交錯式調變碼的比較

圖 3.9 與 3.10 爲多層次編碼的模擬結果與位元交錯式調變碼的比較結果，圖 3.10 的雷利衰退通道爲傑克衰退通道模型所產生，又模擬的假設條件爲表 3.1 所示，且假設信號相位在接收端完全已知。

調變（modulation）	8PSK
標準化都卜勒頻率f_cT （normalized Doppler frequency）	0.01
State 數	8
延遲長度 λ	25
碼字符元/封包 （codeword symbols per package）	667
符元方塊交錯器的交錯深度 L	75

表 3.1 多層次編碼的模擬參數

首先我們來看利用延遲處理器的多層次編碼與位元交錯式調變碼在白色高斯加成性通道上的表現。如圖 3.9 可以看到多層次編碼在 5dB 之後，表現即優於位元交錯式調變碼，試分析其中原因，由於 Hellstern 提出的多層次編碼會利用到先解出的上層位元，再利用集合分割的特性來拉大位元距測的距離；但是以上是假設先解出的上層位元正確，假如上層位元錯誤的話，由於判斷邊界（decision boundary）的改變，所以會影響位元距測的正確性。

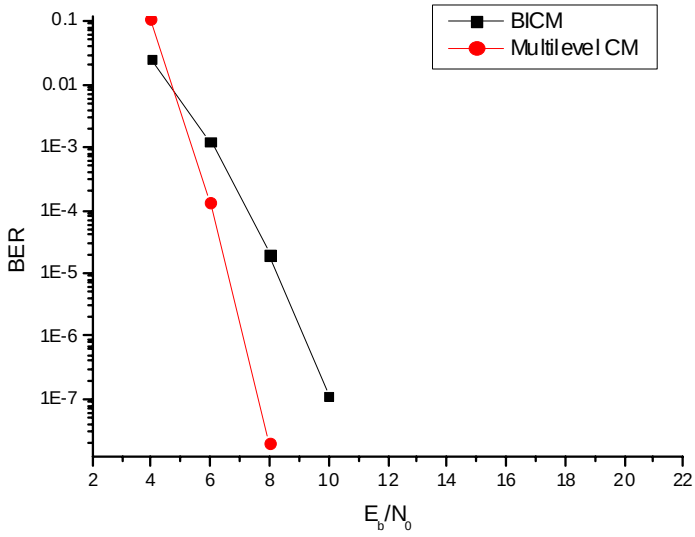


圖 3.9 位元交錯式調變碼與 Hellstern 所提的多層次編碼的次佳解碼方式，在白色高斯加成性通道下的比較圖，其中 BICM 與多層次編碼的限制長度均為 25。

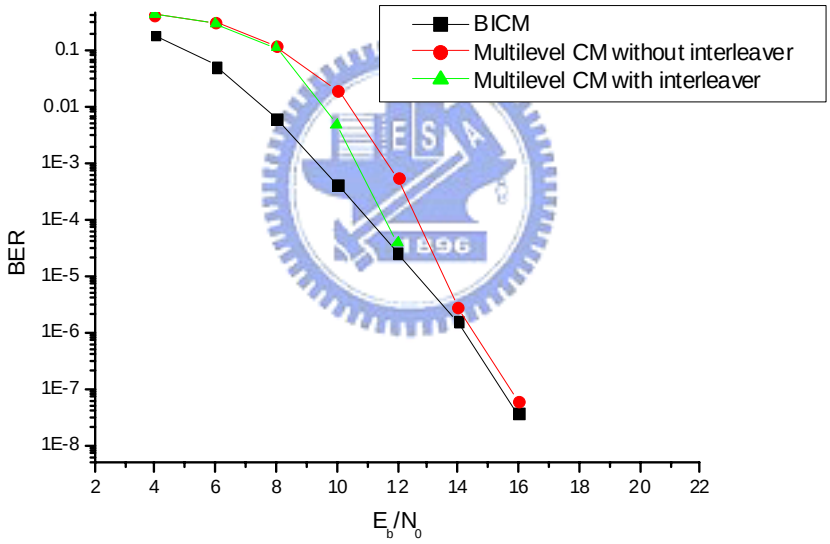


圖 3.10 位元交錯式調變碼與 Hellstern 所提的多層次編碼的次佳解碼方式，在雷利衰退通道下的比較圖，其中 BICM 與多層次編碼的限制長度均為 25。

同樣的道理，也可以在雷利衰退通道上得到驗證。圖 3.10 裡面所用到的交錯器，本篇是採用方塊交錯器（block interleaver），來避免傳輸訊號受到連續的破壞性干擾。關於方塊交錯器的介紹，會在 4.2.3 提到設計的方式。

試著去比較一下兩者的解碼方式便可發現，位元交錯式調變碼的解碼方式並沒有利用到威特比解碼器判斷出來的結果。相對的，由於 Hellstern 所提的多層次編碼架構，利用了延遲處理器來將不同位置的碼字位元做延遲，因此威特比解碼後得到的最大可能性判斷結果（Maximum likelihood decision，ML detection），可以拿來給位元距測計算時當作參考。兩者可以利用的資訊量不同，因此會有如

此的差異。

另外，針對圖 3.10 還有一點值得討論，就是位元交錯式調變碼的信號對應方式。位元交錯式調變碼的對應方式是採格雷碼，假若我們用 3.2.1 介紹的集合分割的觀念來看格雷碼的話，可以發現假如前兩個位元固定的話，例如（000，001）或是（100，101），則 $\Delta(000,001)$ 與 $\Delta(100,101)$ 分別爲 0.586 與 3.414，不管是多少，就是比採用集合分割所得到的 $\Delta_3=4$ 來的小。因此對於本章介紹的多層次編碼而言，在迴授的碼字可靠度高的情形下，（亦即訊號雜音比高的時候），其位元錯誤率會迅速地追上位元交錯式調變碼。

第四章

使用延遲器之首尾相連式的位元交錯調變碼

4.1 簡介

由前兩章，我們可以看到位元交錯式調變碼，雖然在雷利衰退通道下有不錯的表現，但是在白色高斯加成性通道之下的表現並不如傳統的籬柵編碼調變。而 Hellstern 提出的多層次編碼調變，雖然在白色高斯加成性通道之下的表現比傳統的籬柵編碼調變來的好，但是在雷利衰退通道下，卻要在高訊號雜音比的情況下，位元錯誤率才能達到可接受的範圍，而且又有延遲的問題。因此，我們便想到，或許可以利用 Hellstern 的多層次編碼架構，在雷利衰退通道下，將兩者的特性結合起來。

由於在第三章介紹的多層次編碼調變，在解碼時有利用到先前解得的碼字，所以與沒有迴授的位元交錯式編碼比較，先天上就有些許的不公平。於是我們提出了一種位元交錯式調變碼，可以利用第一次解得的碼字串，來讓第二次遞迴解碼時需要參考的碼字串，更具可靠度。而且由於Hellstern的多層次架構下（假設爲 2^m相位鍵移，延遲長度是 λ），每一個封包會必須多傳（m-1） λ 個符元（如圖 4.1），我們想到首尾相連（tail biting）的方法，將多傳的那段（m-1） λ 中的內容，塞在多層次編碼一開始因爲延遲而造成的需要補 0 的區域內。如此一來，雖然一開始由於信號點間的最小距離不大，可能會有較多錯誤位元產生，但是我們希望藉助威特比解碼演算法的錯誤更正能力，能將錯誤的碼字路徑導回正確路徑，使得第一次解碼的結果能具有較高的可信度，來讓第二次遞迴解碼時參考。而在第二次遞迴解碼時，第一次解碼時，一開始解出的那些較不可靠的錯誤位元，便可利用第一次封包末端的解碼結果，得到改善的空間。

本章除了論述我們提出的首尾相連式位元交錯調變碼，也將針對前兩章討論的兩種調變碼，於 4.3 節做彼此的比較與討論。

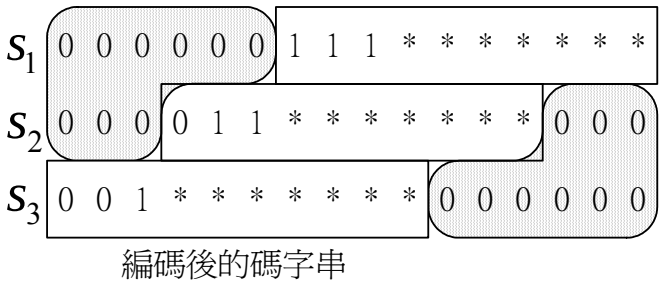


圖 4.1 Hellstern 的多層次編碼架構，可以看到左上與右下的陰影部分，均有非碼字的部分。

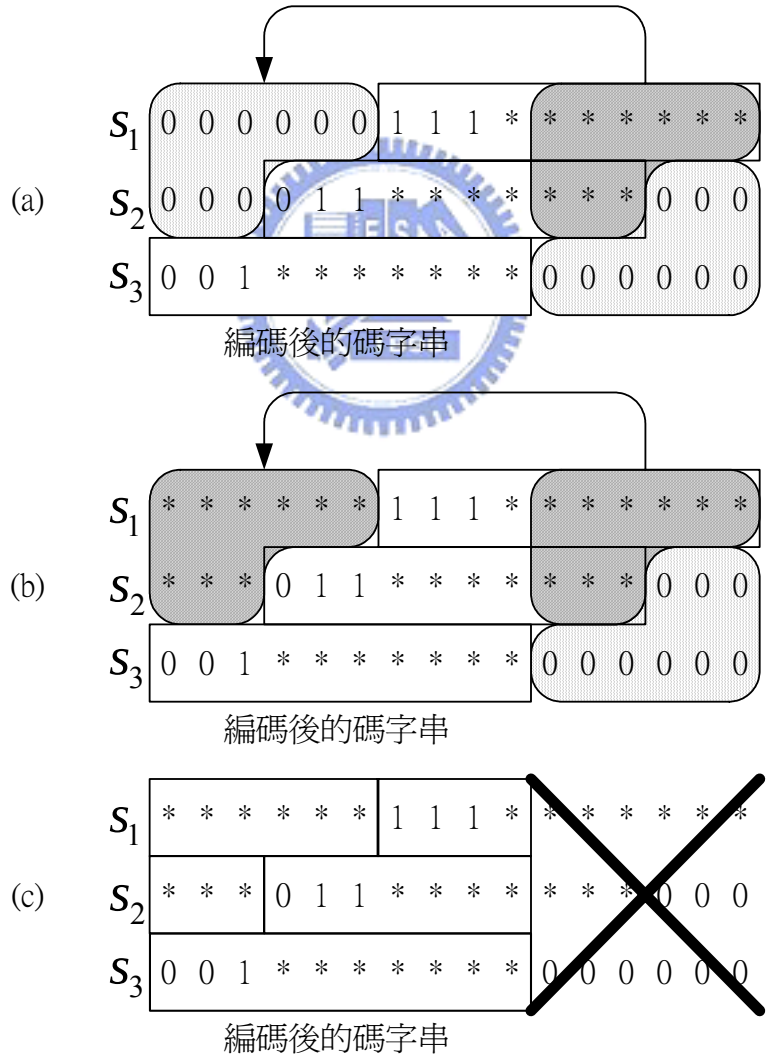


圖 4.2 由(a)→(c)，依序消除掉 Hellstern 的多層次編碼架構因延遲器而造成的多餘的補 0 位元。

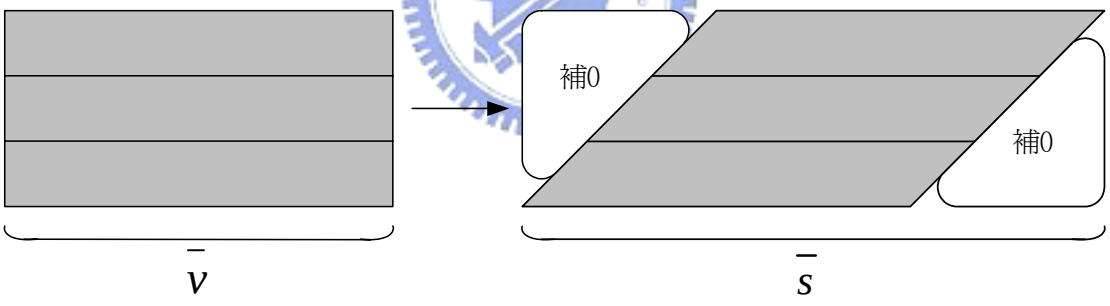
4.2 首尾相連式的位元交錯調變碼

我們提出的架構，其實與 Hellster 所提的多層次編碼相去不遠，只是將傳送封包尾端的符元，如圖 4.2 所示般移到封包的開端，除了可以拉高碼率（code rate），還可以節省傳輸功率的浪費。而且可以由模擬結果發現，在遞迴解碼次數爲 1（也就是只有解碼一次）的情形下，在高訊號雜音比的時候，其位元錯誤率與 Hellstern 所提的多層次編碼結果差不多。而在遞迴解碼次數爲 2 的時候，本篇架構的位元錯誤率已經優於 Hellstern 的表現了。

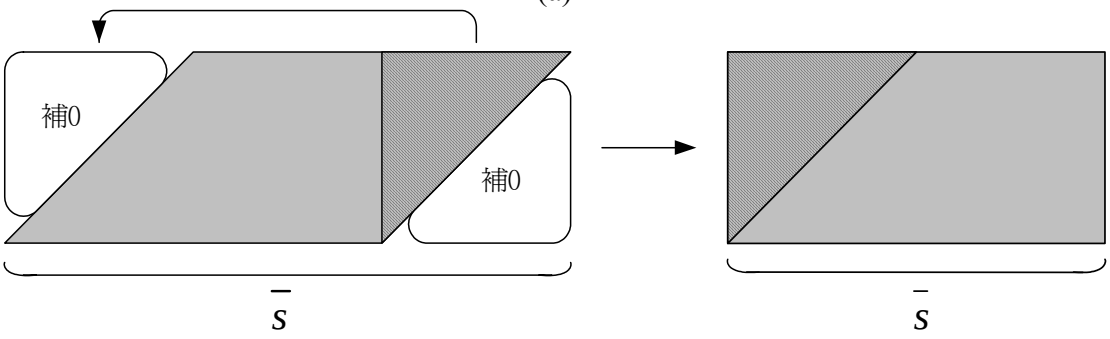
以上是在白色高斯加成性通道上，首尾相連式位元交錯調變碼與 Hellstern 所提的多層次編碼比較的結果。將環境移到雷利衰退通道之中，我們提出的首尾相連式位元交錯調變碼仍然表現優異，若適當地設計交錯器的交錯深度（interleaving depth），使其超過同調時間，便可讓解交錯之後的符元，彼此之間在振幅上沒有相關性，本篇採用方塊交錯器，有關其詳細推導將於 4.2.3 節詳述。

4.2.1 發送器架構

在第三章討論 Hellstern 提出的多層次編碼架構時，我們可以發現延遲處理器可以將碼字串由長方形變成平行四邊行，如圖 4.3(a)所示



(a)



(b)

圖 4.3(a) 延遲處理器造成前後補 0 的示意圖
圖 4.3(b) 末端碼字搬移至前端

因此我們若爲了節省傳輸功率，可以讓$\bar{s}$的封包長度與$\bar{v}$相同，不再多送多餘的 0 位元碼字。方法如圖 4.3(b)所示，將$\bar{s}$封包末端多出的上三角碼字，移至$\bar{s}$封包開端的上三角補 0 處。因此，發送器的架構圖將可表示如圖 4.4。

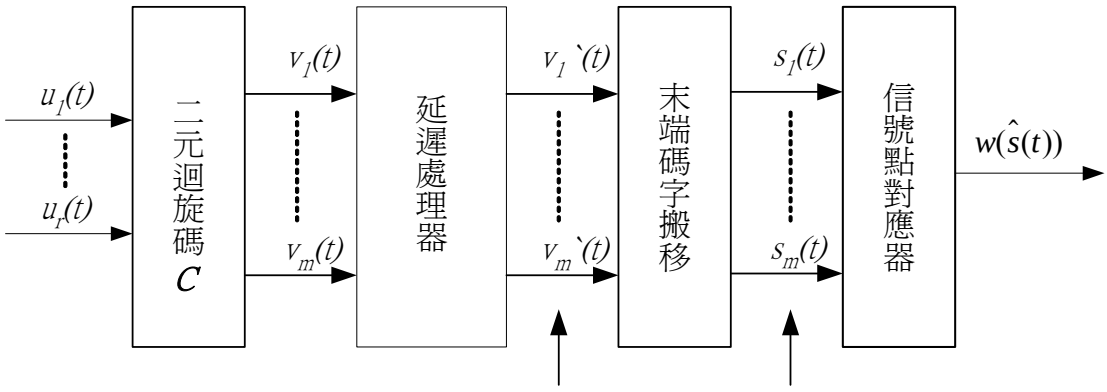


圖 4.4 發送器架構圖

4.2.2 解碼程序

延遲並搬移過後的碼字串$\bar{s}$，與 Hellstern 的多層次編碼並無太大的差異，因此我們仍然可以沿用第三章$\bar{v}$與$\bar{s}$的關係圖。其中只需要略微注意在首尾相連的部分，即$\hat{v}(t),\ t<0$。其原因是因爲在 Hellstern 的多層次編碼中，$t<0$的碼字位元，我們均假設其爲 0 來方便我們做運算；在延遲並搬移之後，$t<0$的碼字便是原本平行四邊行封包$\bar{s}$的末端上三角。在第一次計算的時候，由於我們並沒有$t<0$的資訊。因此，此時的兩兩相對距離，並無法利用到集合分割的特性來拉大距離，位元距測的最小距離均爲Δ_1（參考 3.3 式）。

	$\hat{s}(t)$	$\hat{s}(t+\lambda)$	$\hat{s}(t-(m-j)\lambda)$	$\hat{s}(t+(m-1)\lambda)$
s_1	$v_1(t-(m-1)\lambda)$	$v_1(t-(m-2)\lambda)$	$v_1(t-(j-1)\lambda)$	$v_1(t)$
s_j	$v_j(t-(m-j)\lambda)$	$v_j(t-(m-j-1)\lambda)$	$v_j(t)$	$v_j(t+(j-1)\lambda)$
s_{m-1}	$v_{m-1}(t-\lambda)$	$v_{m-1}(t)$	$v_{m-1}(t+(m-j-1)\lambda)$	$v_{m-1}(t+(m-2)\lambda)$
s_m	$v_m(t)$	$v_m(t+\lambda)$	$v_m(t+(m-j)\lambda)$	$v_m(t+(m-1)\lambda)$

圖 4.5 $\bar{v}$與延遲並搬移過後的$\bar{s}$關係圖

除了封包開頭的上三角部分，我們提出的編碼架構與 Hellstern 是相同的，因此我們一樣可以參考[1]，針對 Hellstern 提出的多層次編碼而提出的次佳解碼方法。解碼步驟與第三章提及的部分相同，便不在此多做論述，比較值得注意的是在封包開頭，以及被搬移到封包開頭的解碼步驟。

同樣的，我們以範例 3.2 來說明。首先我們先假設 $y(t)$是接收端收到加入了通道所給的雜訊的信號，其數學表示式如下

$$y(t) = \rho(t)z(t) + w(t) \qquad (4.1)$$

其中 $\rho(t)$ 代表通道的衰減量，$w(t)$代表白色高斯雜訊。

第一次解碼，且 $0 \le t < 2\lambda$ ：

第一步 我們同樣先針對 $v_j(t) = 0$ 或 1, $j = 1,2,\cdots,m$，來計算第 j 位元的位元距測 $M_{v_j(t)=0}$ 與 $M_{v_j(t)=1}$ 。在計算位元距測 $M_{v_j(t)}$ 時，會利用到 $y(t+\tau_{mj})$ 與先前的解碼結果 $v_i(t-\tau_{ji}),\ 1 \le i < j$ ，只要 $t-\tau_{ji} < 0$ ，也就代表我們利用到被搬移到封包前端的上三角形內的資訊了。由於這是第一次的解碼，所以上三角形內的碼字尚未得知。因此，$M_{v_j(t)}$ 代表著接收訊號 $y(t+\tau_{mj})$ 與有可能的信號點 $w(\hat{s}) = w(s_1, s_2, \cdots, s_m)$ 的距離，其中的 $\hat{s}$ 滿足以下的特性：

$$\begin{cases} s_i \in \{0,1\} & \text{for } i < j \\ s_i = v_j(t) & \text{for } i = j \\ s_i \in \{0,1\} & \text{for } i > j \end{cases} \qquad (4.2)$$

比較（3.8）式與（4.2）式我們可以看到，由於（4.2）式沒有已知的資訊可以利用，我們無法使用到集合分割的特性，因此在每一層，我們的兩兩相對距離都是最短的 $\Delta_1 = 0.586$。第一步的過程可以參考圖 4.6 與 4.7 所示。

第二步 計算分支 $\hat{v}(t) = (v_1(t), v_2(t), \cdots, v_m(t))$ 的分支距測 $M_{\hat{v}(t)}$ 。分支距測 $M_{\hat{v}(t)}$ 的計算方法是累加所有可能的 $M_{v_i(t)},\ i = 1,\cdots,m$ ，也就是

$$M_{\hat{v}(t)=(0,0,0)} = M_{v_1(t)=0} + M_{v_2(t)=0} + M_{v_3(t)=0}$$

$$\vdots$$

$$M_{\hat{v}(t)=(1,1,1)} = M_{v_1(t)=1} + M_{v_2(t)=1} + M_{v_3(t)=1}$$

第三步 將 $M_{\hat{v}(t)}$ 送入威特比解碼器來解迴旋碼 C，因爲其限制長度爲 λ ，所以我們可以解得 $\hat{u}(t-\lambda+1)$ 與 $\hat{v}(t-\lambda+1)$ ，並再將 $\hat{v}(t-\lambda+1)$ 交給下一次使用。

之後的解碼步驟，便與第三章介紹的解碼步驟完全相同，還有最後一段時間的解碼步驟要注意敘述如下

第一次解碼，且 $(N-\lambda) \le t < N$ ：

第一步 與先前的計算方式一樣，我們同樣要針對 $v_j(t) = 0$ 或 1，來計算第 j 位元的位元距測 $M_{v_j(t)=0}$ 與 $M_{v_j(t)=1}$ 。在計算位元距測 $M_{v_j(t)}$ 時，會利用到

$y(t+\tau_{mj})$ 與先前的解碼結果 $v_i(t-\tau_{ji}),\ 1 \le i < j$ 。此時要注意的是，若 $t+\tau_{mj} > N$ ，也就代表我們利用到被搬移到封包前端的接收訊號 $y(t+\tau_{mj})$ ，如圖 4.8 所示。我們以 mod N 來表示 $t+\tau_{mj}$ 除以 N 的餘數。

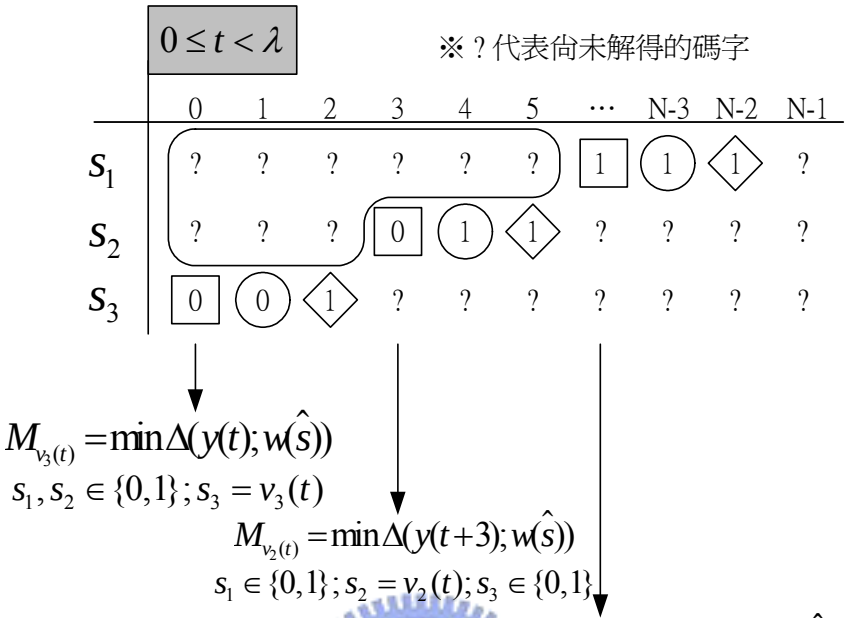


圖 4.6 $0 \le t < \lambda$ 時的 bit metric 計算方式示意圖

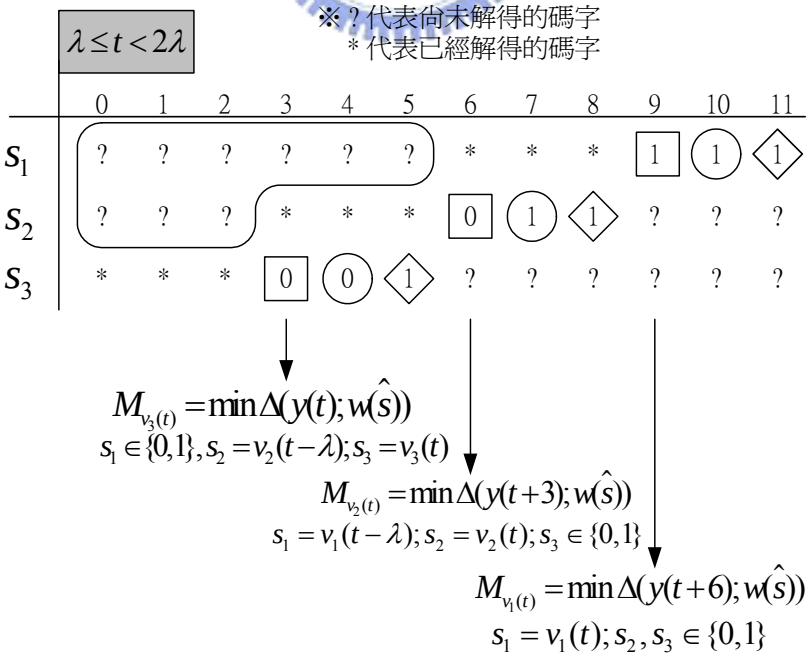


圖 4.7 $\lambda \le t < 2\lambda$ 時的 bit metric 計算方式示意圖。要注意的是此兩段時間可以利用的資訊不一樣多

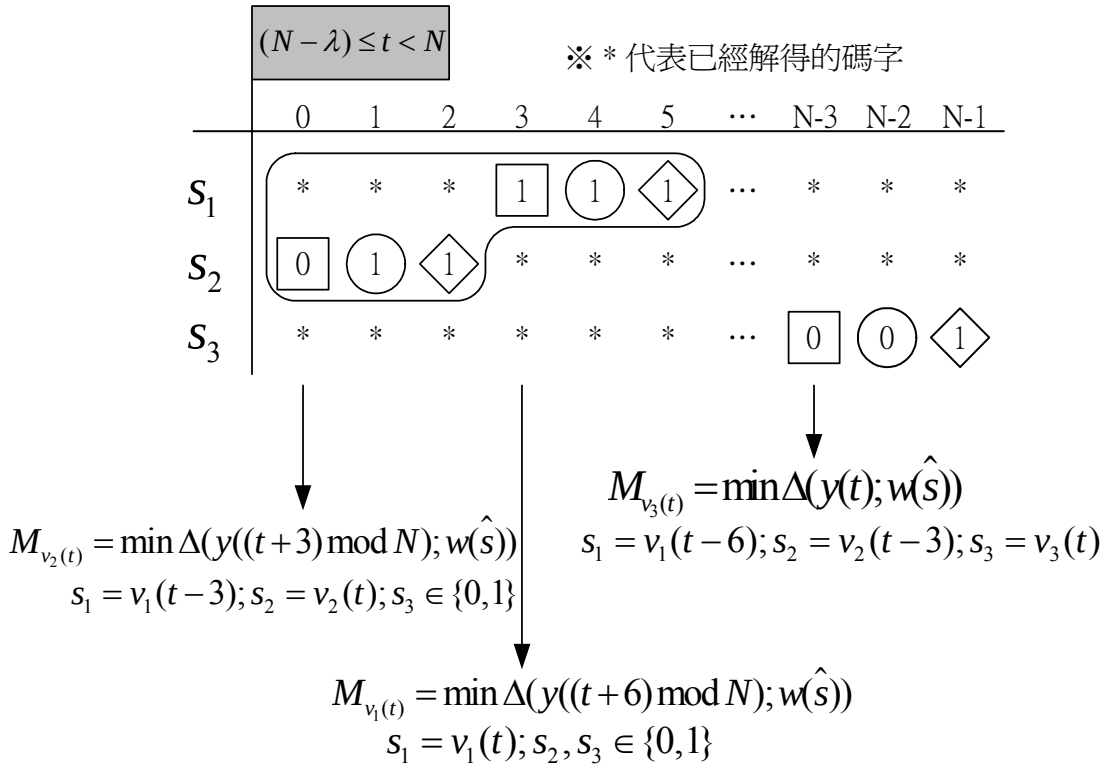


圖 4.8 $(N-\lambda) \le t < N$ 時的 bit metric 計算方式示意圖。

另外，由於爲了要與 Hellstern 所提的多層次編碼做比較，我們試想過將其架構套用在遞迴解碼上，但是由於在第二次解碼時能利用的第一次解碼結果，不會比一開始因延遲器造成的補零來得可靠。因此 Hellstern 所提的多層次編碼，無法用同樣的遞迴解碼次數（2 次以上）與我們所提的首尾相連位元交錯式調變碼比較。

4.2.3 交錯器的設計

交錯器的設計，在遞迴解碼方式中，對於位元錯誤率有很大的影響。一般來說，在遞迴解碼的機制下，交錯器設計的理念，主要有以下兩大設計原則：(1) 增加自由歐基里得距離 (free Euclidean distance) (2) 降低一連串錯誤的可能性。本節所介紹的方塊交錯器，主要是針對第（2）點來做設計。

我們採用方塊交錯器來設計的原因，主要是因爲其交錯次序整齊，並且可以確保連續的符元會在經過交錯後，能落在同調時間之外（coherent time，此處以 Tc表示）。試以圖 4.9 來說明，假設延遲器的延遲長度是 λ，圖 4.9 中 $x(t)$ 爲信號對應器之後的輸出，但還未進入方塊交錯器。我們將 $x(t)$ 橫向輸入方塊交錯器，再縱向地由左而右輸出，交錯後的輸出順序，爲 $x(t)$ ， $x(t+L)$ ， $x(t+2L) \cdots$ $x(t+NL)$ ， $x(t+1) \cdots$。我們希望$N+1 > T_c$，這樣才不至於使 $x(t)$ 和 $x(t+1)$ 的振幅衰減具相關性。

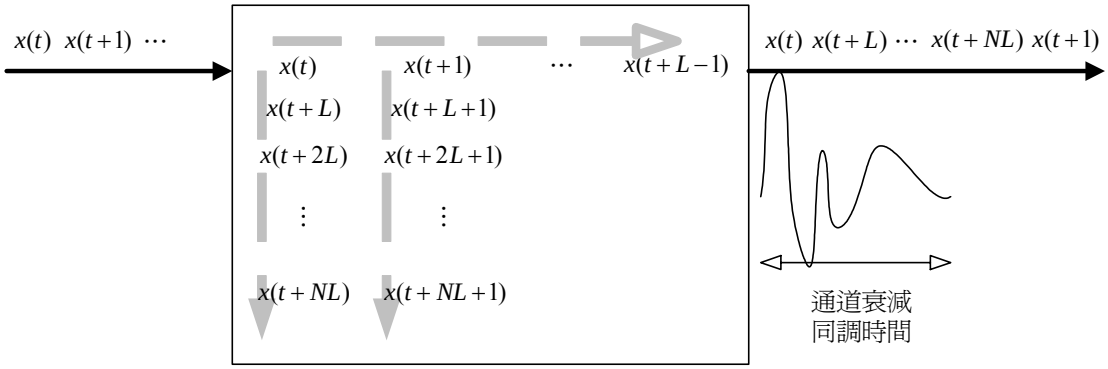


圖 4.9 發送端傳送的碼字符元，經方塊交錯器交錯後的時序分佈圖

但是別忘了，接收端收到訊號之後，必須延遲 $m\lambda$-1 之後才能得到解碼結果 $v(t)$，因此我們希望在這段時間內，經交錯還原後的每個接收訊號的振幅衰減均不具相關性，所以方塊交錯器還需多加一道限制：

$$t+L>t+(m\lambda-1) \quad \rightarrow \quad L>(m\lambda-1)$$

因此，方塊交錯器的每一個橫列，其長度L必須大於$m\lambda$-1，而且高度的大小$N+1$ 必須超過T_c。

4.3 模擬結果與討論

圖 4.10 與 4.11 爲首尾相連式調變碼針對遞迴解碼次數的模擬結果，圖 4.11 的雷利衰退通道爲傑克衰退通道模型所產生，又模擬的假設條件爲表 4.1 所示，且假設信號相位在接收端完全已知。

調變（modulation）	8PSK
標準化都卜勒頻率f_cT （normalized Doppler frequency）	0.01
State 個數	8
延遲長度 λ	25
碼字符元/封包 （codeword symbols per package）	667
符元方塊交錯器的交錯深度 L	75

表 4.1 首尾相連式調變碼的模擬參數

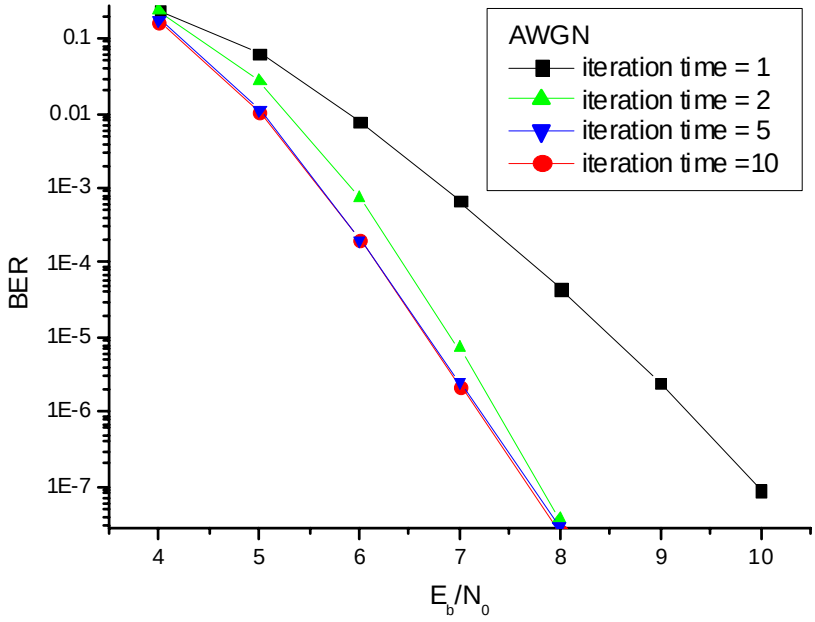


圖 4.10 在白色高斯加成性通道下，隨著遞迴次數不同的位元錯誤率

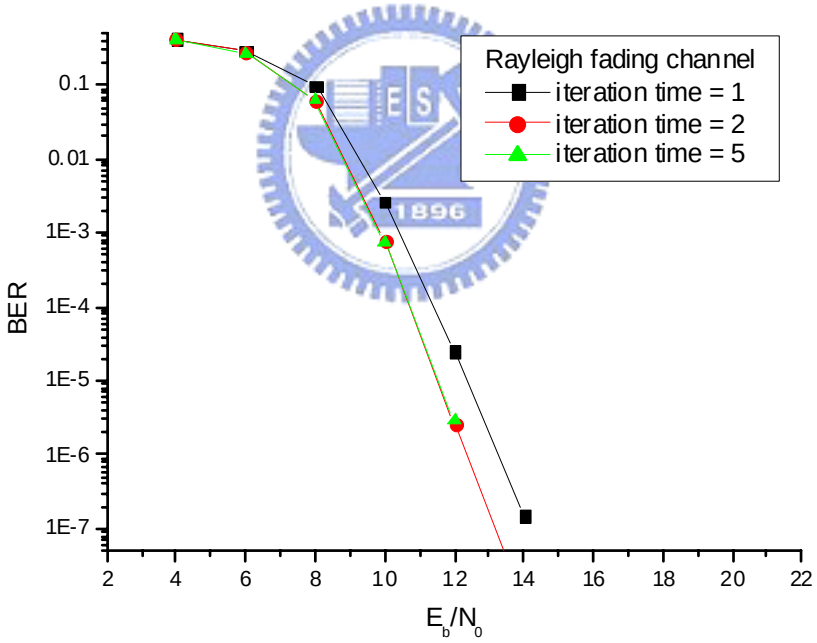


圖 4.11 在雷利衰退通道下，隨著遞迴次數不同的位元錯誤率

由圖 4.10 與圖 4.11 我們可以看到，隨著遞迴解碼的次數增加，在白色高斯加成性通道與雷利衰退通道，其位元錯誤率均會隨著下降。但是我們可以發現，在解碼次數超過 2 之後，再怎麼做遞迴解碼，位元錯誤率下降的程度幾乎已經可說是零了。這或許是因爲我們所提的架構裡，是採用硬式判斷迴授（hard decision feedback）的關係，使得改善空間並不如預期的多。

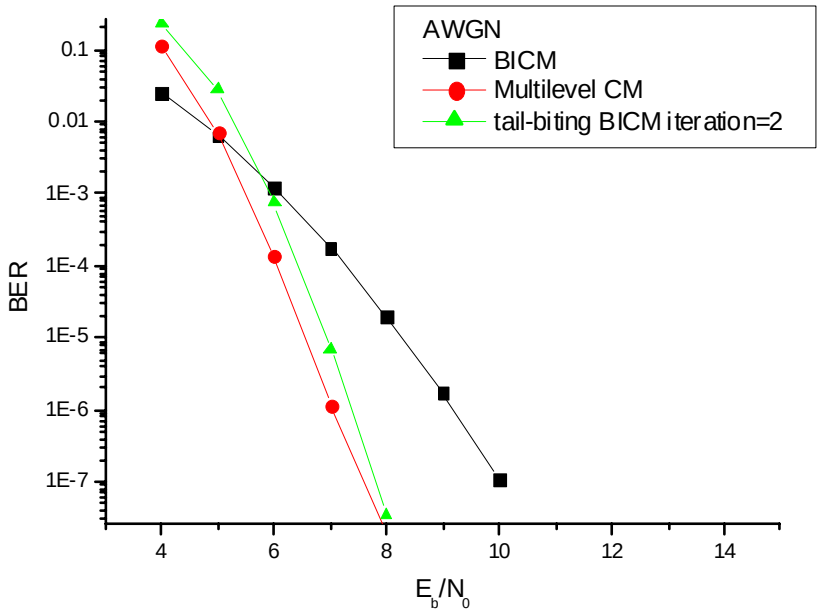


圖 4.12 在白色高斯加成性通道下，有關本篇介紹的三種不同調變碼，其位元錯誤率

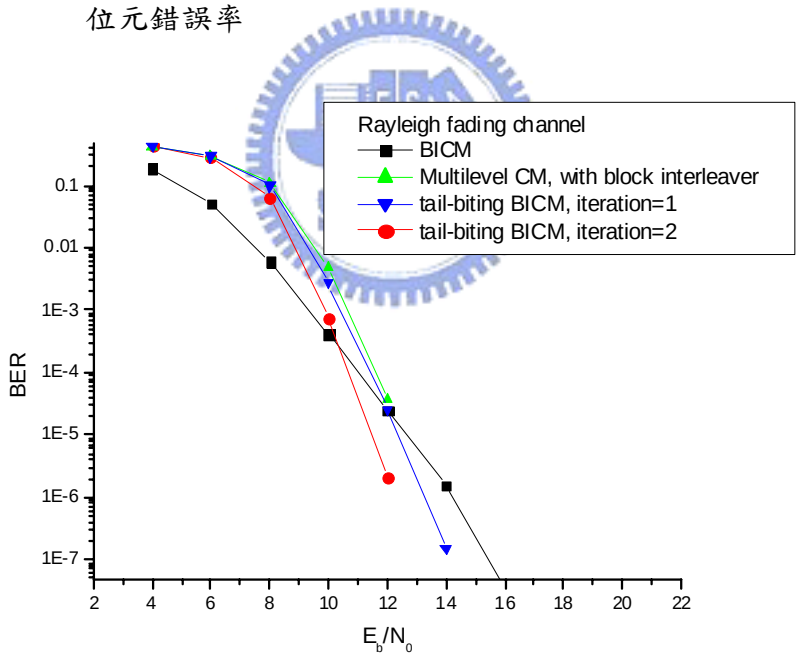


圖 4.13 在雷利衰退通道下，三種不同調變碼之位元錯誤率表現。

由圖 4.12 與圖 4.13 我們可以看到，在環境爲白色高斯加成性通道時，本篇提出的首尾相連位元交錯式調變碼，與 Hellstern 所提的多層次編碼一直有一小段差距，直到訊號雜音比在 8 dB 時才追上，這原因應該是因爲在首尾相連式的『尾巴』與『頭』（即原本平行四邊行的首尾兩端）相連處，第一次解碼所得的尾巴再怎麼準確，也不會比『接收端已知上層位元一定是零』來得準。至於在雷

利衰退通道部分，卻可以看到雖然使用用同樣的方塊交錯器，但是 Hellstern 所提的多層次編碼在表現上一直略遜於首尾相連式，原因是首尾相連式可以做遞迴解碼。

4.4 本章結論

本章中我們所提出的首尾相連式位元交錯調變碼，將封包末端碼字部分挪到前端的部分來傳送，除了可以節省傳輸功率之外，更可以利用首尾相連的部分來達成遞迴解碼的動作。在運算量上面來看，與其他的遞迴解碼機制相比，[1]提出的此種次佳解碼方式又不會太複雜。

以 8PSK 爲例，$\boldsymbol{m}=3$，除了分支距測在威特比解碼時所需的計算量之外，[1]提出的次佳解碼方式還需要計算位元距測，以 $v_3(t)$ 來說，此時由於已經得知 $s_1 = v_1(t-2\lambda)$ ， $s_2 = v_2(t-\lambda)$ ，我們只需要針對 $v_3(t)=0$ 或 1 來計算個別的位元距測，並不需要多餘的計算；而在 $v_2(t)$ 時，我們已知的只有 $s_1 = v_1(t-\lambda)$ ，而針對 $v_2(t)=0$ 或 1 來計算位元距測，但在 $s_3 \in \{0,1\}$ 的部分需要一個比較器來找出 $y(t)$ 離（ $s_1 = v_1(t-\lambda), v_2(t)$, 0 或 1）最近的點，因此在 $v_2(t)$ 的部分我們需要 2・1＝2 個比較器；在 $v_3(t)$ 時，同樣的，針對 $v_1(t)=0$ 與 1，我們需要 2・（4－1）＝6 個比較器。另外，三個位元距測均求得後，在計算每一種可能的分支距測時，還需要 $\boldsymbol{m}$-1 個加法器。因此我們可以得知，此種解碼方式比一般的編碼調變所需的計算量，只多出了 $2(1+3+\cdots(2^{m-1}-1))=2(2^{m-1}-\boldsymbol{m}-1)$ 個比較器與 $2^m(m-1)$ 個加法器。

而且在解碼次數超過 2 之後，由圖 4.10 與 4.11 便可得知不用再繼續遞迴解碼下去，較諸其他的遞迴解碼機制動輒 4、5 次以上才能達到收斂值，算是此種首尾相連式位元交錯調變碼的優點。

第五章

結論

5.1 主要成果與比較

知道我們提出的首尾相連式位元交錯調變碼，在白色高斯加成性通道之下，與 Hellstern 的多層次編碼不相上下，更優於傳統的位元交錯式調變碼。將通道環境換成雷利衰退通道時，其位元錯誤率的表現，在訊號雜音比超過某個程度之後，即優於傳統的位元交錯式調變碼了。

那麼與其他遞迴性解碼比較的結果差異如何呢？我們找了 2002 年由 Xiaodong Li, Aik chindapol, and James A. Ritcey 發表的一篇論文來做比較 [4]，如圖 5.1 所示，爲位元交錯式調變碼 - 遞迴解碼（BICM-Iterative Decoding）的架構。可以看到，其架構圖內迴授的資訊，爲軟式判斷（soft decision）的結果，也就是並不會對該次的計算結果，做出最大可能性的判斷，而是做最大後置機率的判斷（Maximum a posteri probability decision，MAP decision）。相對的，本篇的迴授內容，爲威特比解碼器對輸入信號的硬式判斷（hard decision feedback）所做出最大可能性的判斷（ML detection）。於是我們可以知道本篇雖然運用了判斷迴授（decision feedback）的機制，但是迴授內容與判斷方式上的差異，會造成位元錯誤率上如圖 5.2 與 5.3 的差距。

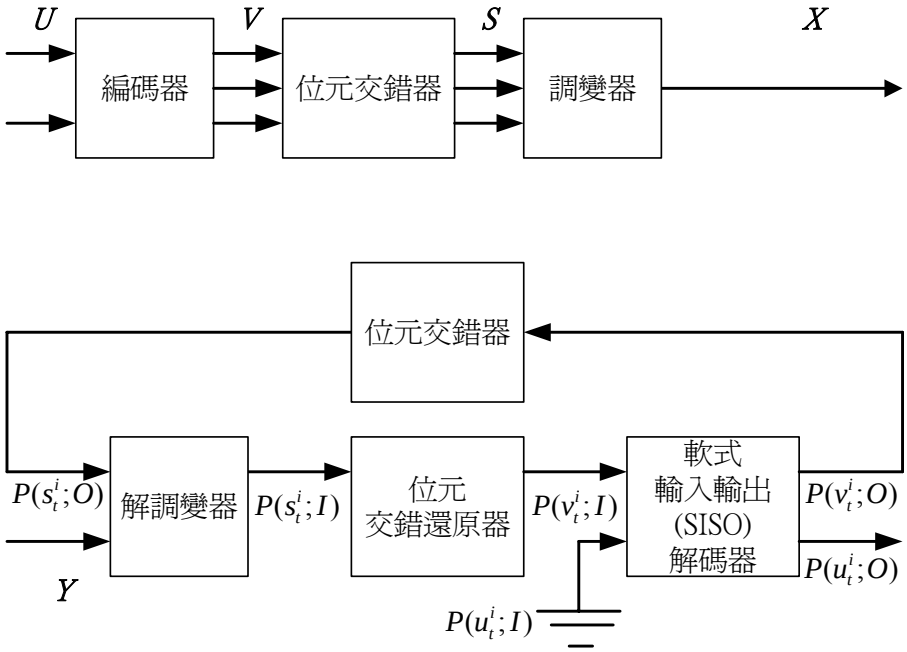


圖 5.1 位元交錯式調變碼 - 遞迴解碼的方塊圖

圖 5.1 中的 $P(q;I)$與 $P(q;O)$，分別代表著這次所傳的資訊是前置機率（a priori probability）與及後置機率（a posteriori probability），

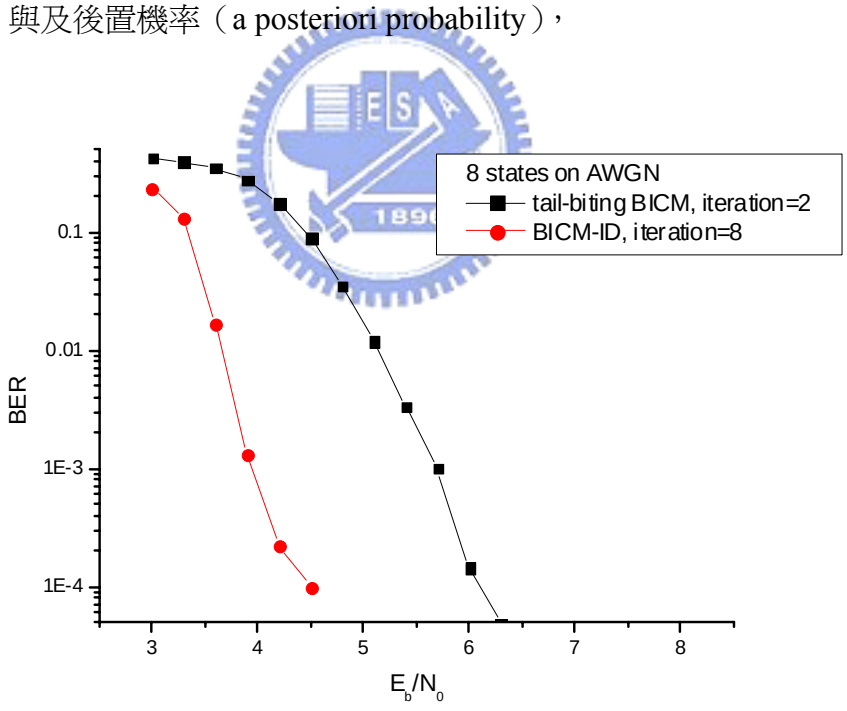


圖 5.2 在白色高斯加成性通道下，與 BICM-ID 比較的結果

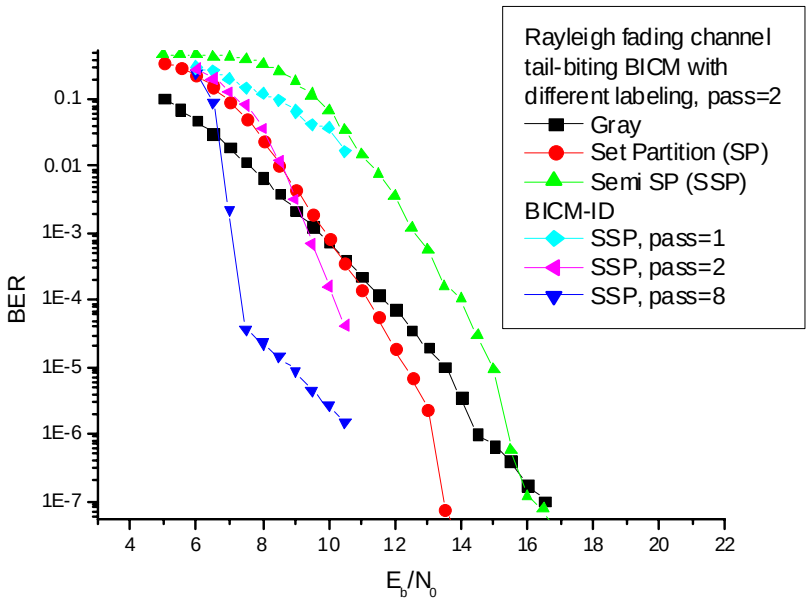


圖 5.3 在雷利衰退性通道下，與 BICM-ID 比較的結果

[4]用所謂的半集合分割（Semi Set Partitioning，SSP）來做信號點對應。從圖 5.4 可以看到，灰色區塊分散的程度，以半集合分割最爲分散，格雷碼最集中，所以在第一次解碼時，由於事先並無資訊的獲得，集合分割的特性無法拉大信號點間的最小距離，因此在位元錯誤率的表現上以格雷碼對應方式最佳。

但是當遞迴遞廻解碼次數增加後，我們獲得了較可靠的資訊，便可將原本的最短距離 Δ_1 拉開到 Δ_2 甚至 Δ_3 。圖 5.4 中的紅色線段代表除了該位元未知，其他位元均爲已知情況下，位元距測的最小距離。在遞迴解碼機制下，紅色線段越長，代表判斷結果可以幫助提升的最短距離越大。另一方面。我們可以看到格雷碼在這方面的距離，隨著遞迴解碼次數增加，增益的幅度並沒有其他兩者來的大。至於爲何[4]爲何使用半集合分割此種信號對應方式，是因爲平均來看，半集合分割的平均最小距離是三者中最大的。而本篇的首尾相連式位元交錯調變碼，以及 Hellstern 的多層次編碼，因爲使用的解碼方式均爲王佳盈[1]提出的次佳解碼，利用到的迴授碼字位元，均位於未確定位元的上層，即（3.4）式所描述的，並沒有如[4]一般利用到下層的碼字位元資訊，因此使用平均最小距離的半集合分割，並不會比集合分割來的好，而且因爲半集合分割的灰色區域太過分散，第一次解碼時得到的解碼結果，可信度將比其他兩種信號對應器來的低。

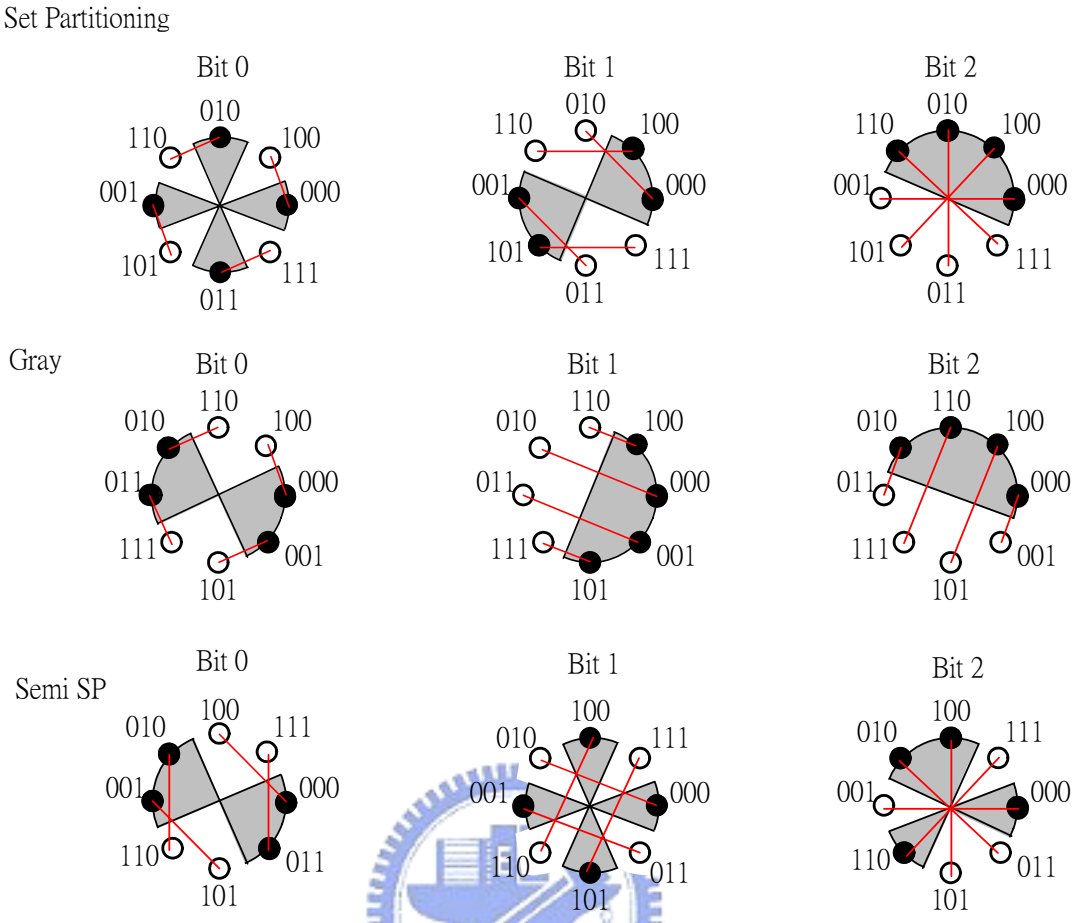


圖 5.4 三種信號對應方式的比較圖。

5.2 後續研究

我們提出的解碼架構，雖然硬式判斷迴授在遞迴解碼次數超過 2 便遇到了瓶頸。但也由於是利用硬式判斷的結果，因此解碼時的運算量（只根據上層的解碼結果，來計算該位元的位元距測）與複雜度（硬式判斷明確地判定出是 0 或是 1，軟式判斷則還需要對此判定加上相關的可靠性，例如"0"的可靠性是 70%，"1"可靠性是 30%）相對的低了許多，能得到如圖 5.2、5.3 的結果，已屬不錯。

因此，我們可以嘗試著套用如[4]的軟式判斷迴授（soft decision feedback），或是以帶有更多迴授資訊的硬式判斷迴授（比如說，迴授的碼字位元不止 0 與 1 的二選一，再加入介於 0 與 1 之間的灰色模糊地帶的判斷），以及 Hellstern 的多層次編碼架構這兩個主軸，來進行遞迴式解碼，應該可以得到更好的位元錯誤率；或是可以由交錯器的設計來著手，或許會有比方塊交錯器更爲適合這種使用延遲器的多層次編碼架構。另外，多層次編碼架構利用遞迴式解碼時每一層的最小距離事實上是不一樣的，因此很適合來從事不等效保護編碼（unequal error protection coding），這些課題都值的進一步研究。

參考文獻

[1] Jia-Yin Wang, “Multilevel Coding for Large Free Distances 具大自由距離之多層次編碼,”

[2] E. Zehavi, “8-PSK trellis codes for a rayleigh channel,” *IEEE Trans. Commun.*, vol. 40, pp. 873-884, May 1992.

[3] G. Caire, G. Taricco, and E. Biglieri, “Bit-interleaved coded modulation,” *IEEE Trans. Inform. Theory*, vol. 44, pp. 927-946, May 1998.

[4] Xiaodong Li, Aik Chindapol, and James A. Ritcey, “Bit-interleaved coded modulation with iterative decoding and 8PSK signaling,” *IEEE Trans. Commun.*, vol. 50, pp. 1250-1257, Aug. 2002.

[5] Gunter Hellstern, “Coded Modulation with Feedback Decoding Trellis Codes,” *IEEE Conference on Commun.*, pp. 1071-1075, 1993.

[6] Xiaodong Li, James A. Ritcey, “Bit-Interleaved Coded Modulation with Iterative Decoding,” *IEEE Conference on Connun*., pp. 858-863, vol. 2, 1999.

[7] Xiaodong Li, James A. Ritcey, “Trellis-Coded Modulation with Bit Interleaving and Iterative Decoding,” *IEEE J. Select. Areas Commun.* pp. 715-724, vol. 17, No. 4, April 1999.

中英譯名對照表

Added White Gaussian Noise：白色高斯加成性雜訊
a priori probability：前置機率
a posteriori probability：後置機率
Bit-Interleaved Coded Modulation，BICM：位元交錯式調變碼
BICM-Iterative Decoding：位元交錯式調變碼 - 遞迴解碼
Bit metric：位元距測
Bit position：位元位置
Block interleaver：方塊交錯器
Branch metric：分支距測
Burst error：訊號叢錯
Code rate：碼率
Codeword：碼字
Codeword path：碼字路徑
Codeword symbol：碼字符元
Coherent time：同調時間
Constraint length：限制長度
Decision feedback：判斷迴授
Decision boundary：判斷邊界
Delay processor：延遲處理器
Diversity gain：多向度增益
Free distance：自由距離
Free Euclidean distance：自由歐幾里得距離
Gray code：格雷碼
Hard decision：硬式判斷
Interleaver：交錯器
Interleaving depth：交錯深度
Iterative decoding：遞迴式解碼
Iterative gain：遞迴（式解碼）增益
Level：層次
Maximum a posteri probability decision，MAP decision：最大後置機率的判斷
Maximum likelihood decision，ML detection：最大可能性判斷
Maximum-likelihood sequence estimation：最大可能性串列估計
Mean fade duration：平均衰退區間長度
Memoryless：無記憶性

Memoryless channel：無記憶性通道
Metric：距測
Minimum free distance：最小自由距離
Multilevel Coding 多層次編碼
Multiple phase shift keying, MPSK：多相鍵移
Nonoverlap：不重疊
Pairwise distance：兩兩相對距離
Partition：分割
Partition chain：分割鏈
PSK：相位鍵移
Rayleigh fading channel：雷利衰退通道
Quadrature amplitude modulation, QAM：二維振幅調變
Set partitioning：集合分割
Semi Set Partitioning，SSP：半集合分割
Signal mapper：信號點對應器
Signal constellation：信號星座集
Slow fading：慢速衰退
Soft decision：軟式判斷
Soft decision feedback：軟式判斷迴授
Subsets：子集合
Survival metric：殘存距測
Symbol：符元
Tail biting：首尾相連
TCM，Trellis Coded Modulation：籬柵式編碼調變
Unequal error protection coding：不等效保護編碼
Viterbi decoding algorithm：威特比解碼演算法

作者簡介

湯秉熹，男，高雄市人，1977 年生於高雄市。

1989 年　畢業於高雄市七賢國小。

1992 年　畢業於高雄市七賢國中。

1996 年　畢業於高雄市立高雄高級中學。

2000 年　畢業於國立成功大學電機工程學系。

2005 年 4 月獲得碩士學位，投入職場。

Graduate Course

1. Digital Communication
2. Random Process
3. Digital Signal Process
4. Detection and Estimation
5. Special Topics on Digital Communication (Digital Communication Receivers)
6. Special Topics on Digital Communication (Array Signal Process)
7. Queueing Theory
8. Spread Spectrum Communication