行政院國家科學委員會專題研究計畫　成果報告

寬頻無線通訊系統的錯誤控制機制之設計(3/3)

計畫類別：個別型計畫
計畫編號：NSC93-2213-E-009-021-
執行期間：93 年 08 月 01 日至 94 年 07 月 31 日
執行單位：國立交通大學電信工程學系(所)

計畫主持人：蘇育德

報告類型：完整報告

處理方式：本計畫可公開查詢

中　華　民　國 94 年 10 月 25 日

中文摘要

蘋狄(Pyndiah)氏所發展的渦輪解碼演算法是乘積碼的基礎，此演算法適用於任何以線性區塊碼構成的乘積碼。這個渦輪式解碼的乘積碼，現在通常被稱為渦輪乘積碼 (TPC) 或是區塊渦輪碼 (BTC) ，有著與其他解碼法不同的優點如解碼法簡單，快速收斂，高碼率 (code rate) ，以及非常低甚至沒有錯誤極限 (error floor)等。雖然其在瀑布區 (waterfall region)的表現不如傳統渦輪碼(turbo codes) 。

本計畫探討了結合區塊渦輪碼及正交分頻多工信號的可能性。我們將部分重點放在涉及到主控了系統錯誤率表現之主要的接收機設計，亦即通道估測與解碼演算法。我們檢測結合了重複估測及解碼近似的表現。至於傳送的波形設計上，我們檢驗了交錯器 (interleaver) 的作用且提出一種新型的區塊渦輪碼，我們稱之為平行串接乘積碼 (PCPC)。平行連鎖乘積碼有類似傳統渦輪碼的結構，然而是用含資料部分的乘積碼所構成。我們也證明了，我們所採用的交錯器的確有助於增加平行串接乘積碼之最小翰明距離。模擬的數值結果也告訴我們，與近似碼率之對應的區塊渦輪碼相比，平行連鎖乘積碼的確有比較好的性能表現。

關鍵詞：正交分頻多工，渦輪乘積碼，交錯器，通道估測。

Abstract

A benchmark in the development of product code is the invention of the iterative (turbo) decoding algorithm by Pyndiah. Pyndiah's algorithm works for any product code using linear block component codes. The turbo decoded product codes, now often referred to as turbo product codes (TPC) or block turbo codes (BTC), have the distinct merits of simple decoding, fast convergence, high code rate and very low (or no) error floor although their performance at the waterfall region is not as impressive as that of conventional turbo codes.

This report summarize our effort in the past year that explores the feasibility of using BTC in conjunction with the orthogonal frequency division multiplexing (OFDM) signals. We focus on the major receiver design concerns that dominate the error rate performance of the system, namely, the channel estimation and decoding algorithm. We examine the performance of a joint iterative channel estimation and decoding approach. As for the transmitted waveform design, we check the impact of the interleaver and present a new class of BTC that we refer to as parallel concatenated product codes (PCPC). The PCPC has a structure similar to the conventional turbo code with the component codes replaced by systematic product codes. Some of its algebraic properties and related decoding issues are investigated. Numerical results indicate that it does outperform its BTC counterpart with a comparable code rate.

Keyword: OFDM, TPC, interleaver, channel estimation.

Contents

List of Figures

Chapter 1

Introduction

Since its invention in 1993, the turbo code [9] has revolutionized the field of error-correcting codes. Over the years, we have come to realize that the extraordinary performance offered by such a simple-looking encoding structure is brought about by the efficient iterative decoding algorithm and the random-like interleaver. The iterative algorithm provides a message-exchange mechanism that collects extrinsic information from related samples to help enhancing the decision reliability. The interleaver makes sure that the range of message exchange is as large as possible and that short cycles in the associated bi-partie graph be eliminated or the number of these cycles be minimized.

The original turbo code uses two identical recursive systematic convolutional codes as its component codes. The idea of using block codes as component codes was first proposed by Pyndiah [2] who realized that a class of block codes called product codes has inherent parallel concatenation feature with a simple block interleaver. Pyndiah [2] then applied the concept of iterative (turbo) decoding to decode a product code. To distinguish this approach from the classic turbo code, product codes using iterative decoding scheme are referred to as block turbo codes (BTC) or turbo product codes (TPC).

The report concentrates on the designs of some receiver baseband subsystems of an orthogonal frequency division multiplexing (OFDM) system that employs a BTC. In

particular, we consider the issues of channel estimation, tracking and BTC decoding. Our approach follows the popular turbo paradigm, applying the turbo principle to BTC decoding as well as channel estimation.

To employ turbo decoding, soft output must be generated by the constituent block decoder. When the component codes are BCH or RS codes, to reduce decoding complexity, instead of MAP algorithm, one might use the Chase decoder [7] in conjunction with extended Euclid's algorithm [5] to produce reliability estimates associated with hard-decision decoded bits or symbols. Using this hybrid algorithm, iterative decoding of product codes becomes possible although one can only expect suboptimal performance and iterative decoding gain is more significant at high SNR region. On the other hand, since the minimum distance of a typical BTC is usually much larger than that of a classic turbo code, the associated error performance curve has a much lower error floor, if it does exist.

Cyclic product codes were first introduced by Burton and Weldon in 1965 [1]. These codes enjoy the implementation advantages of cyclic codes and, in addition, possess some important structural properties:

1. Conditions are given which ensure that the product of two, three or arbitrarily finite many cyclic codes is itself a cyclic code.

2. Cyclic product codes are shown to be capable of correcting of both bursts and random errors.

3. The generator polynomial of the cyclic product code is derived and shown to be a simple function of the generator polynomials of the constituent codes.

4. The minimum distance of the resulting code is the product of those of the constituent codes.

In order to achieve efficient forced erasure decoders, Hirst *et al.* [3] re-order the Chase algorithm's repeated decodings such that the inherent computational redundancy is

greatly reduced without degrading performance. The result is a highly efficient Fast Chase implementation.

Another key issue related to our investigation is that of channel estimation. We adopt a model-based approach as it is more robust in time-varying fading channels and requires less channel state information like the channel correlation matrix and the average signal-to-noise ratio (SNR) per bit. A two-dimensional ($2D$) model-based channel estimation is used in our receiver. The $2D$ model exploits the frequency and time domain correlation so that better channel estimate is obtained. The channel estimation algorithm consists of four steps. By dividing the time-frequency plane into sub-blocks, and with the received samples at the pilot locations, the algorithm first estimates the coefficients associated with a $2D$ surface model by employing least square (LS) fit on the pilots in each sub-block. Once the model coefficients are known, the frequency responses at non-pilot locations can then be computed.

To further enhance the performance of BTCs, we propose a new class of turbo-like codes called parallel concatenated product codes. PCPC improves the minimum distance property while retaining the merit of low decoding complexity of product codes. We prove that using the Fibonacci interleaver does help increasing the minimum distance. The regularity of the interleaver also reduces the implementation complexity and makes parallel interleaving feasible. A decoding method based on modified Pyndiah algorithm and a proper scheduling is suggested. Numerical examples indicate that the new PCPC outperforms Pyndiah's product code with comparable code rate.

In summary, this report investigates the feasibility and performance of a product coded OFDM system with simple iterative joint channel estimation and decoding algorithms. In chapter 2, we provide an elementary introduction to TPC, followed by a brief discourse on the OFDM scheme in Chapter 3. The OFDM channel estimator and an iterative decoding algorithm used are presented in Chapter 4 where we also introduce the helical interleaving scheme and explain its effectiveness. In Chapter 5, the class of of *par-*

allel concatenated product codes and related properties are presented. Related decoding algorithm and its numerical performance examples are given. Finally, Chapter 6 draws some concluding remarks and suggests a few research topics for further investigations.

Chapter 2

Introduction to Turbo Product Codes

This chapter provides some background material for the class of product codes, its constituent codes and related decoding methods.

2.1 Product codes

The concept of product codes is very simple and relatively efficient for building very long block codes by using two or more short block codes. Fig. 2.1 shows a typical (two-dimensional) product coding scheme that arranges the information symbols in a rectangular array and encodes each row and column individually by two linear block codes $\mathcal{C}^1$ and $\mathcal{C}^2$. The resulting augmented rectangular array of Fig. 2.1 form a codeword of the product code $\mathcal{C}^1 \otimes \mathcal{C}^2$ with rows and columns being $\mathcal{C}^1$ and $\mathcal{C}^2$ codewords.

In summary, given two systematic linear block codes $\mathcal{C}^1$ and $\mathcal{C}^2$ with parameter (n_1, k_1, δ_1) and (n_2, k_2, δ_2), the product code $\mathcal{P} = \mathcal{C}^1 \otimes \mathcal{C}^2$ is obtained (see Fig.2.1) by

1. placing $(k_1 \times k_2)$ information bits in an array of k_1 rows and k_2 columns.
2. coding the k_1 rows using code $\mathcal{C}^2$.
3. coding the n_2 columns using code $\mathcal{C}^1$.

The parameters of the product code $\mathcal{P}$ are thus given by $n = n_1 \times n_2$, $k = k_1 \times k_2$ and $\delta = \delta_1 \times \delta_2$, and the code rate $\mathcal{R}$ is given by $\mathcal{R} = \mathcal{R}_1 \times \mathcal{R}_2$, where $\mathcal{R}_i$ is the code rate of code $\mathcal{C}^i$.

The resulting code has a minimum distance equals to the product of the minimum distances of the constituent codes. Therefore, one can build very long block codes with large minimum Hamming distance by combining short codes with small minimum Hamming distance. Note that each information symbol in the rectangular is encoded by both $\mathcal{C}^1$ and $\mathcal{C}^2$ codes and is related to different sets of information symbols. Such a encoding scheme is similar to that of classic turbo codes and, as a result, can be iteratively (turbo) decoded. When a turbo-like decoder is used, the class of product codes is referred to as block turbo code (BTC) or turbo product code (TPC).

In contrast to the classic convolutional turbo codes (CTC), BTC has the distinct attractive feature that high rate and large minimum distance codes can be easily found. There is no need of special interleaver design and it is not necessary to search for favorable puncture patterns. In case elementary linear codes such as Hamming or extended Hamming codes are used as the constituent codes then the corresponding decoding complexity is far less than that conventional CTC and the convergence rate is often faster. Moreover, because of parallel independent encoding, the decoding procedure could be carried out in parallel and high decoding throughput is readily realizable.

2.2 A brief review of BCH codes

Using binary BCH codes instead of Hamming codes gives us a more flexible range of choices in code rate R, codeword length and minimum distance.

An (n, k, δ) binary BCH code has codeword length n, k information bits per codeword, and minimum Hamming distance $2t+1 = \delta$, respectively. Its generator polynomial $g(x)$ of degree $r = n - k$ is an irreducible factor of $x^n - 1$, $n = 2^q - 1$ for some q, with $t - 1$ consecutive powers of a primitive element of GF(2^q) as its zeros.

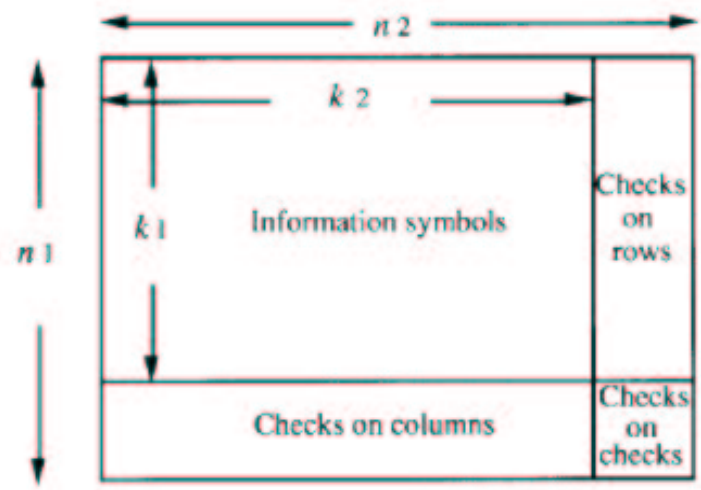


Figure 2.1: A two-dimensional product code $\mathcal{P} = \mathcal{C}^1 \otimes \mathcal{C}^2$. (reprinted from [8])

2.2.1 Systematic encoder for BCH codes

Consider the the binary shift-register (SR) based BCH encoder shown in Fig. 2.2. Let a k-symbol message block $\mathbf{m} = (m_0, m_1, \ldots, m_{k-1})$ be associated with the message polynomial $m(x) = m_0 + m_1 x + \cdots + m_{k-1}x^{k-1}$, and the n-symbol codeword $(c_0, c_1, \ldots, c_{n-1})$ be associated with the codeword polynomial $c_m(x) = c_0 + c_1 x + \cdots + c_{n-1}x^{n-1}$. Then a systematic encoding procedure can be described as follows.

1. *Step 1. Multiply the message polynomial $m(x)$ by x_{n-k}.*

2. *Step 2. Divide the result of Step 1 by the generator polynomial $g(x)$. Let $p(x)$ be the remainder.* Polynomial division is performed through the use of a linear feedback shift register (LFSR). It divides $a(x) = a_0 + a_1 x + \cdots + a_{n-1}x^{n-1} = m_0 x^{n-k} + \cdots + m_{k-1}x^{n-1}$ by $g(x) = g_0 + g_1 x + \cdots + g_{r-1}x^{r-1} + x^r$ and retains the remainder $p(x) = p_0 + p_1 x + \cdots + p_{r-1}x^{r-1}$. The symbols $a_0, a_1, \ldots, a_{n-2}, a_{n-1}$ are fed into the shift register one at a time in order of decreasing index. When the last symbol (a_0) has been fed into the rightmost SR cell, the SR cells will contain the coefficients of the remainder polynomial.

3. *Step 3.* *Set* $c(x) = x^{n-k}m(x) - p(x)$. The code word output is thus given by $(c_{n-1}, c_{n-2}, \ldots, c_1, c_0) = (m_{k-1}, m_{k-2}, \cdots, m_0, p_{n-k-1}, p_{c-k-2}, \cdots, p_0)$.

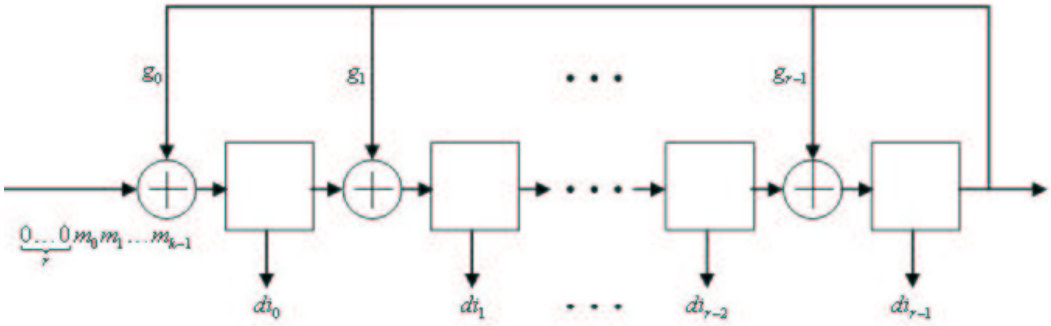


Figure 2.2: Shift-register division of $a(x)$ by $g(x)$

Fig. 2.3 shows a example of a (7,4,3) BCH code. The information bits is 1010 and the output code word is 1010011.

2.2.2 Extended Euclidean algorithm

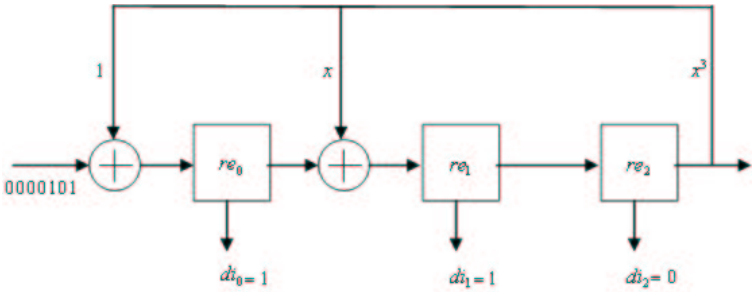
Let $\alpha^1, \alpha^2, \cdots, \alpha^{\delta-1}$ be the $(\delta - 1)$ roots of $g(x)$, where α is a primitive nth root of unity, and denote by $r(x)$ the received polynomial associated with the received vector $\mathbf{R} = (r_0, r_1, \cdots, r_{n-1}) = \mathbf{C} + \mathbf{E}$, where $\mathbf{E} = (e_0, e_1, \cdots, e_{n-1})$ is the error vector whose nonzero entries are at the i_lth positions, $l = 1, \cdots, v, v \leq t$. We further define the syndrome generating polynomial by $S(x) = \sum_j S_j x^j$, where $S_j = r(\alpha^j)$ is the jth syndrome, and the error locator polynomial by

$$\Lambda(x) = \prod_{k=1}^{v}(1 - X_k x) = \Lambda_0 + \Lambda_1 x + \Lambda_2 x^2 \cdots + \Lambda_v x^v, \tag{2.1}$$

where $X_k = \alpha^{i_k}$ are error locations. Then one can show that

$$\Lambda(x)[1 + S(x)] \equiv \Omega(x) \ (mod\, x^{2t+1}) \tag{2.2}$$

where $\Omega(x)$ is the error-evaluator polynomial of degree $v-1$. The above identity is called the *key equation* for decoding BCH codes. Once we solve the *key equation*, finding $\Lambda(x)$



(a)

SR cells	re_0	re_1	re_2	
Input $a_6 = 1$	1	0	0	
Input $a_5 = 0$	0	1	0	
Input $a_4 = 1$	1	0	1	
Input $a_3 = 0$	1	0	0	
Input $a_2 = 0$	0	1	0	
Input $a_1 = 0$	0	0	1	
Input $a_0 = 0$	1	1	0	
Final state	1	1	0	$\Leftrightarrow di(x) = x+1$

(b)

Figure 2.3: (a)Shift-register division of $1 + x + x^4 + x^6$ by $1 + x + x^3$ (b)Shift-register cell contents during division of $1 + x + x^4 + x^6$ by $1 + x + x^3$

and $\Omega(x)$ that satisfy it, then the error locations can be found by solving $\Lambda(x)$ (via the so-called *Chien Search*) and the corresponding error magnitudes (for non-binary codes) can be solved by using the so-called Forney's algorithm which involves evaluating both $\Lambda(x)$ and $\Omega(x)$.

2.2.3 Decoding BCH codes

The Extended Euclidean algorithm is a simple (but not the most efficient) BCH/RS decoding algorithm. It operates on two elements $(\hat{a}, \hat{b})$ from an Euclidean Domain E at a time. Given the initial conditions $\hat{r}_{-1} = \hat{a}, \hat{r}_0 = \hat{b}, \hat{s}_{-1} = 1, \hat{s}_0 = 0, \hat{t}_{-1} = 0, \hat{t}_0 = 1$, it

proceeds according to the following set of recursion relations.

$$\begin{aligned}\hat{r}_i &= \hat{r}_{i-2} - \hat{q}_i\hat{r}_{i-1},\\ \hat{s}_i &= \hat{s}_{i-2} - \hat{q}_i\hat{s}_{i-1},\\ \hat{t}_i &= \hat{t}_{i-2} - \hat{q}_i\hat{t}_{i-1}.\end{aligned} \tag{2.3}$$

where $\hat{r}_i, \hat{s}_i, \hat{t}_i, \hat{q}_i$ are all elements of E. The algorithm terminates when the remainder $\hat{r}_n = 0$. The remainder $\hat{r}_{n-1}$ is then the GCD of $\hat{a}$ and $\hat{b}$. The recursion relations insure that, at any given point in the algorithm, we have the relation

$$\hat{s}_i\hat{a} + \hat{t}_i\hat{b} = \hat{r}_i \tag{2.4}$$

The reason why the Extended Euclidean Algorithm can be used to decoding BCH codes can be readily answered by first noting that the *key equation* implies

$$\Theta(x)x^{2t+1} + \Lambda(x)[1 + S(x)] = \Omega(x) \tag{2.5}$$

and if $\Omega'(x)$ is the GCD of the two polynomials x^{2t+1} and $1+S(x)$ then there exist $\Lambda'(x)$ and $\Theta'(x)$ such that

$$\Theta'(x)x^{2t+1} + \Lambda'(x)[1 + S(x)] = \Omega'(x) \tag{2.6}$$

It can be proved that the error-evaluator polynomial $\Omega(x)$ and the error-locator polynomial $\Lambda(x)$ are proportional to $\Omega'(x)$ and $\Lambda'(x)$, respectively. Hence one can apply the Euclidean algorithm with the finite-degree polynomials over GF(2) as the Euclidean Domain of concern to x^{2t+1} and $1 + S(x)$. The pair $(t_j, r_j) \stackrel{def}{=} (\Lambda^j(x), \Omega^j(x))$ for some proper j will then our solution. The particular solution that corresponds to the error locator and magnitude polynomials is obtained when $\Omega^j(x)$ has degree less than or equal to that of $\Lambda^j(x)$. The decoding steps is summarized in the following steps.

D1 *Compute the syndromes* $S_i, i = 1 \sim 2t$ *and form the syndrome polynomial* $S(x)$.

D2 *Set the following initial conditions:* $\hat{r}_{-1}(x) = x^{2t+1}, \hat{r}_0(x) = 1 + S(x), \hat{t}_{-1}(x) = 0, \hat{t}_0(x) = 1$.

D3 *Using the extended algorithm, compute the successive remainders* $\hat{r}_i(x)$ *and the corresponding* $\hat{t}_i(x)$ *until the following stopping condition is reached:* $deg[\hat{r}_i(x)] \leq t$.

D4 *Find the roots of* $\hat{t}_i(x) = \Lambda(x)$*, thus determining the error locations.*

D5 *Determine the magnitude of the errors.*

Example 1. *Suppose that we have received the vector* $\tilde{\mathbf{r}} = (1011011)$. *The received polynomial is then given by* $\tilde{r}(x) = 1 + x + x^3 + x^4 + x^5$. *Obviously. Applying the Extend Euclidean Algorithm with the initial condition* $\hat{r}_{-1}(x) = x^3$ *and* $\hat{r}_0(x) = 1 + S(x) = 1 + \alpha^3 x + \alpha^5 x^2$. *Following the above five steps, we obtain* $\Lambda(x) = \alpha^5 + \alpha x$, *and find its root at* α^{-3} *(see Fig. 2.4). Since it is a binary code we can just reverse the polarity of the error position* $\tilde{r}_3$, *changing it back to* 0.

j	$\hat{r}_j$ $\Omega^{(j)}(x)$	$\hat{q}_i$ $q_j(x)$	$\hat{t}_i$ $\Lambda^{(j)}(x)$
-1	x^5	——	0
0	$\alpha^6 x^2 + \alpha^3 x + 1$	——	1
1	α^5	$\alpha x + \alpha^5$	$\alpha x + \alpha^5$

Figure 2.4: Example of single error correction using Extended Euclidean algorithm

2.2.4 Iterative decoding of product codes

As suggested by Pyndiah, a product code can be iteratively (turbo) decoded. There are a variety of soft-decision decoding algorithms for block codes. The Chase algorithm and its variations offer a good balance and tradeoff between complexity and performance. List-decoding often consists of two stages: (i) finding candidate codewords based on the received samples and (ii) generating soft output. As in a turbo decoder, the soft output

is then passed to the ensuing decoding round as the extrinsic (a priori) information. We describe the decoding algorithm for a linear block code as follows.

2.2.4.1 Selecting candidate codewords

Suppose an (n, k, δ) linear block code $\mathcal{C}$ is BPSK-modulated and transmitted over an additive white Gaussian noise (AWGN) channels. Denote by $\boldsymbol{X} = (x_1, \cdots, x_l, \cdots, x_n)$ and $\boldsymbol{R} = (r_1, \cdots, r_l, \cdots, r_n)$ the transmitted codeword and received vector, where $x_l \in \{+1, -1\}$. Then $\boldsymbol{R} = \boldsymbol{X} + \boldsymbol{E}$, where the noise vector is $\boldsymbol{E} = (e_1, e_2, \cdots, e_n)$ in which e_i are i.i.d. zero-mean Gaussian random variables with variance σ^2. Denote by $\boldsymbol{A} = (a_1, \cdots, a_l, \cdots, a_n)$ the a priori information of the codeword bits where

$$a_l = \ln \frac{\Pr\{c_l = +1\}}{\Pr\{c_l = -1\}}. \tag{2.7}$$

Following the spirit of Chase-II list decoding algorithm [7], we suggest the following three-step algorithm

A1. Use $\boldsymbol{R}$ and $\boldsymbol{A}$, if available, to obtain the hard decision vector $\boldsymbol{Y} = (y_1, y_2, \cdots, y_n)$ as well as their reliability (extrinsic information). Determine the $p = \lfloor \delta/2 \rfloor$ positions associated with the least reliable binary elements of $\boldsymbol{Y}$, where the reliability of y_j is given by $\Lambda(x_j)$ and is related to the log-likelihood ratio (LLR)

$$\Lambda(x_j) = \ln \frac{\Pr\{r_j | x_j = +1\}}{\Pr\{r_j | x_j = -1\}} + a_j = \frac{2}{\sigma^2} r_j + a_j \tag{2.8}$$

A2. Bit-flipping the most unreliable p positions on $\boldsymbol{Y}$ to form the set of 2^p test patterns $\boldsymbol{T}^q$ $(0 \le q \le 2^p - 1)$.

A3. Form test sequence $\boldsymbol{Z}^q$ where $z_l^q = y_l \oplus t_l^q$ and decode $\boldsymbol{Z}^q$ using an algebraic (or hard) decoder and add the decoded codeword $\boldsymbol{C}^q$ to subset Ω. Decision $\boldsymbol{D} = (d_1, d_2, \cdots, d_n)$ of a row (or column) of the product code is then obtained by applying decision rule:

$\boldsymbol{D} \in \Omega$ is a local maximum likelihood codeword if

$$|\boldsymbol{R} - \boldsymbol{D}|^2 \leq |\boldsymbol{R} - \boldsymbol{C^i}|^2 \ \forall \ \boldsymbol{C^i} \in \Omega. \tag{2.9}$$

where

$$|\boldsymbol{R} - \boldsymbol{C^i}|^2 = \sum_{l=1}^{n} \frac{(r_l - c_l^i)^2}{2\sigma^2} - \frac{c_l^i}{2} a_l \tag{2.10}$$

is the metric between $\boldsymbol{R}$ and $\boldsymbol{C^i}$.

2.2.4.2 Soft output generation

The reliability of decision d_j about the transmitted symbol x_j, given the observation $\boldsymbol{R}$, is

$$\Lambda'(x_j) = \ln \left(\frac{\Pr\{x_j = +1 | \boldsymbol{R}\}}{\Pr\{x_j = -1 | \boldsymbol{R}\}} \right) \tag{2.11}$$

Let $S_j^{+1} \subset \boldsymbol{C}$ be the set of codewords whose jth coordinate c_j^i is $+1$ and $S_j^{-1} \subset \boldsymbol{C}$ be the set of codewords with -1 in their jth coordinate. Then we have

$$\Pr\{x_j = +1 | \boldsymbol{R}\} = \sum_{\boldsymbol{C^i} \in S_j^{+1}} \Pr\{\boldsymbol{X} = \boldsymbol{C^i} | \boldsymbol{R}\} \tag{2.12}$$

$$\Pr\{x_j = -1 | \boldsymbol{R}\} = \sum_{\boldsymbol{C^i} \in S_j^{-1}} \Pr\{\boldsymbol{X} = \boldsymbol{C^i} | \boldsymbol{R}\}, \tag{2.13}$$

and (2.11) becomes

$$\Lambda'(x_j) \quad = \ln \left(\frac{\sum\limits_{\boldsymbol{C^i} \in S_j^{+1}} \mathrm{p}\{\boldsymbol{X} = \boldsymbol{C^i} | \boldsymbol{R}\}}{\sum\limits_{\boldsymbol{C^i} \in S_j^{-1}} \mathrm{p}\{\boldsymbol{X} = \boldsymbol{C^i} | \boldsymbol{R}\}} \right) \tag{2.14}$$

$$\approx \ln \left(\frac{\sum\limits_{\boldsymbol{C^i} \in S_j^{+1} \cap \Omega} \mathrm{p}\{\boldsymbol{X} = \boldsymbol{C^i} | \boldsymbol{R}\}}{\sum\limits_{\boldsymbol{C^i} \in S_j^{-1} \cap \Omega} \mathrm{p}\{\boldsymbol{X} = \boldsymbol{C^i} | \boldsymbol{R}\}} \right) \tag{2.15}$$

$$\approx \ln \left(\frac{\max\limits_{\boldsymbol{C^i} \in S_j^{+1} \cap \Omega} \mathrm{p}\{\boldsymbol{X} = \boldsymbol{C^i} | \boldsymbol{R}\}}{\max\limits_{\boldsymbol{C^i} \in S_j^{-1} \cap \Omega} \mathrm{p}\{\boldsymbol{X} = \boldsymbol{C^i} | \boldsymbol{R}\}} \right) \tag{2.16}$$

At high SNRs, (2.16) can be approximated by

$$\Lambda''(x_j) = |\boldsymbol{R} - \boldsymbol{C}^{-1(j)}|^2 - |\boldsymbol{R} - \boldsymbol{C}^{+1(j)}|^2 \tag{2.17}$$

where $C^{+1(j)} \in S_j^{+1} \cap \Omega$ and $C^{-1(j)} \in S_j^{-1} \cap \Omega$ are the codewords with the minimum metric and one of $C^{+1(j)}$ and $C^{-1(j)}$ is $\boldsymbol{D}$. Substituting (2.10) into (2.17), we obtain

$$\Lambda''(x_j) = \frac{2r_j}{\sigma^2} + a_j + \sum_{l=1, l\neq j}^{n} \left(\frac{2r_l}{\sigma^2} + a_l \right) c_l^{+1(j)} p_l \tag{2.18}$$

where

$$p_l = \begin{cases} 0, & \text{if } c_l^{+1(j)} = c_l^{-1(j)} \\ 1, & \text{if } c_l^{+1(j)} \neq c_l^{-1(j)} \end{cases} . \tag{2.19}$$

If there is a competing codeword, the soft output corresponding to d_j is

$$\Lambda''(x_j) = \frac{2r_j}{\sigma^2} + a_j + w_j \tag{2.20}$$

and the extrinsic information is

$$w_j = \sum_{l=1, l\neq j}^{n} \left(\frac{2}{\sigma^2} r_l + a_l \right) c_l^{+1(j)} p_l. \tag{2.21}$$

$$= \Lambda''(x_j) - \frac{2r_j}{\sigma^2} - a_j \tag{2.22}$$

Before we demonstrate the extrinsic information of position without competing codeword, we define a coordinate set

$$V = \{ \, j \mid \exists \; \boldsymbol{C}^i \in \Omega \; s.t. \; c_j^i \neq d_j \}. \tag{2.23}$$

Thus the mean value $\overline{w}$ of the extrinsic information $w_j, \; j \in V$ can be computed by

$$\overline{w} = \frac{1}{|V|} \sum_{j \in V} |w_j|. \tag{2.24}$$

The extrinsic information for a position without competing codeword can be

$$w_j = \beta \times \overline{w}, \; j \notin V, \tag{2.25}$$

where β is a constant varying with iterations and will be given in the next section.

With the extrinsic information derived above, the iterative decoding can apply the information as the a priori information of the next APP decoding run.

2.3 Turbo decoding of product code

Let us consider the decoding of the rows and columns of a product code $\mathcal{P}$ transmitted over a Gaussian channel by QPSK signaling. On receiving matrix $[\boldsymbol{R}]$ corresponding to a transmitted codeword $[\boldsymbol{X}]$, the first decoder performs the soft decoding of the rows (or columns) of $\mathcal{P}$ using as input matrix $[\boldsymbol{R}]$. Soft-input decoding is performed using the Chase algorithm and extrinsic information $[\boldsymbol{W}(2)]$ is computed using (2.22) or (2.25), where index 2 indicates that we are considering the extrinsic information for the second decoding of $\mathcal{P}$ which was computed during the first decoding of $\mathcal{P}$. The soft input for the decoding of the columns (or rows) at the second decoding of $\mathcal{P}$ is given by

$$[\boldsymbol{R}(2)] = [\boldsymbol{R}] + \alpha(2)[\boldsymbol{W}(2)] \tag{2.26}$$

where $\alpha(2)$ is a scaling factor which takes into account the fact the standard deviation of samples in matrix $[\boldsymbol{R}]$ and in matrix $[\boldsymbol{W}]$ are different (see [9] and [10]). Besides, this scaling factor α is also used to reduce the effect of the extrinsic information in the soft decoder in the first decoding steps when the BER is relatively high. It takes a small value in the first decoding steps and increases as the BER tends to zero. The decoding procedure described above is then generalized by cascading elementary decoders illustrated in Fig. 2.5. Several *rules of thumb* for turbo decoding product codes obtained

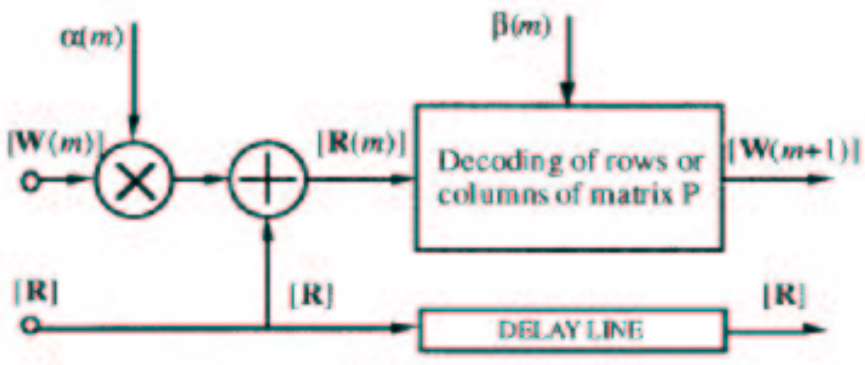


Figure 2.5: Block diagram of elementary block turbo decoder. (reprinted from [8])

through numerical experiments are [8]

1. *Test sequences:* The number of test patterns is 16 and are generated by the four least reliable bits ($p = 4$).

2. *Weighting factor α:* To reduce the dependency of α on the product code, the mean absolute value of the extrinsic information $|w|$ derived using (2.24) is normalized to one. The evolution of α with the decoding number is

$$\alpha(m) = [0.0, 0.1, 0.2, 0.3, 0.4, 0.6, 0.8, 1.0]. \tag{2.27}$$

3. *Reliability factor β:* To operate under optimal conditions, the reliability factor should be determined as a function of the BER. For practical considerations, we have fixed the evolution of β with the decoding step to the following values

$$\beta(m) = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.8, 1.0]. \tag{2.28}$$

In the first decoding step β is set to 20% of the mean of the normalized extrinsic information computed using (2.24) and is gradually increased to 100%. Note that experimental results indicate no significant performance degradation if values of β is modified by $\pm 10\%$.

Chapter 3

Orthogonal Frequency Division Multiplexing Modulation

OFDM is the effective transmission technology to combat frequency-selective fading. By inserting a guard interval larger than expected maximum multipath delay to a regular OFDM symbol, such that the inter-symbol interference (ISI) caused by multipath propagation can be eliminated in the receiving end by discarding the guard interval part. An one-tap compensation in the frequency domain is all one needs to demodulate the transmitted data. Of course, OFDM also has some shortcomings such as its sensitive to frequency offset and phase distortion.

3.1 OFDM Basics

3.1.1 Transmitter and receiver

A frequency domain OFDM signal is shown as in Fig. 3.1. Each subcarrier can be modulated by signals from different constellations such as BPSK, QPSK, 16-QAM etc. Denoted by N, T, $f_k = \frac{k}{T}$ and X_k the number of subcarriers, the carrier frequency, the symbol duration and the modulated symbol at the kth subcarrier, where $0 \leq k < N$.

We express an OFDM symbol within the interval $[0.T]$ as

$$s(t) = \begin{cases} \sum_{k=\frac{-N}{2}}^{\frac{N}{2}-1} X_k \phi_k(t) & , 0 \leqslant t \leqslant T \\ 0 & , \text{otherwise} \end{cases} \tag{3.1}$$

$$= \left(\sum_{k=\frac{-N}{2}}^{\frac{N}{2}-1} X_k \phi_k(t) \right) u_T(t), \tag{3.2}$$

where $\phi_k(t) = \frac{1}{\sqrt{T}} e^{j2\pi f_k t}$ is the kth subcarrier and

$$u(t) = \begin{cases} 1 & , 0 \leqslant t \leqslant T \\ 0 & , \text{otherwise} \end{cases} \tag{3.3}$$

is a time domain windowing function. Each OFDM symbol contains subcarriers that are nonzero over a T-second interval. Hence, the spectrum of a single symbol is a convolution of a group of Dirac pulses located at the subcarrier frequencies with the spectrum of a square pulse of duration T seconds, $sinc(\pi f T)$, which is equal to zero for all frequencies f that are an integer multiple of $1/T$.

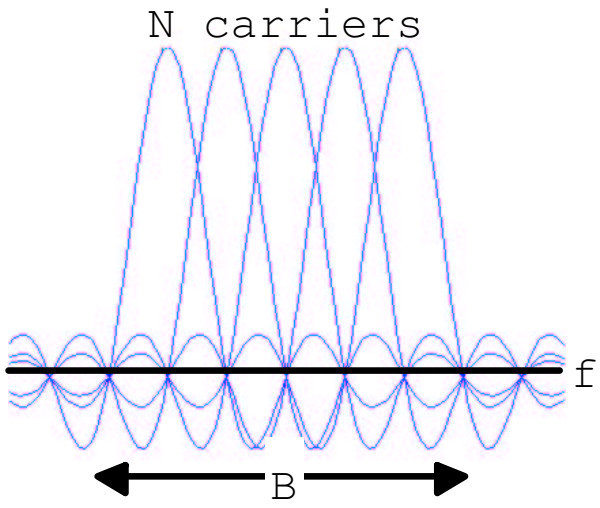


Figure 3.1: An OFDM signal with N subcarriers and bandwidth B Hz. (reprinted from [11])

Data transmitted over a multi-carrier channel can be arranged as a rectangular array

shown in Fig. 3.2, which reminds us of a TPC codeword. For every symbol time, one column of the modulated symbols $\{X_k\}$ are sent.

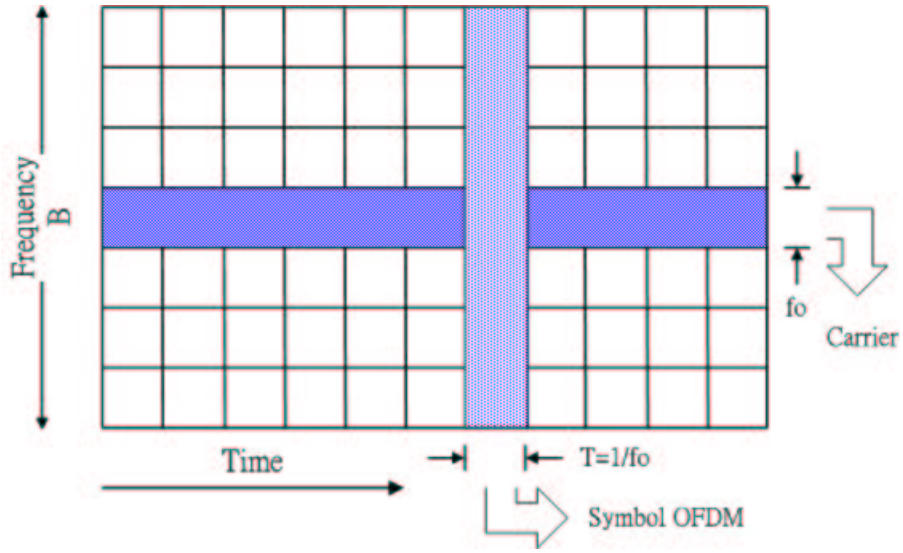


Figure 3.2: A two-dimensional view of a multi-carrier signal; the inter-carrier separation f_0 = any integer multiple of $\frac{1}{T}$. (reprinted from [11])

A schematic structure of OFDM modulator and demodulator are plotted in Fig. 3.3. Denoted by $r(t) = s(t) + n(t)$ the received signal where $n(t)$ is the complex additive white Gaussian random process. Projecting $r(t)$ into the jth subcarrier subspace

$$Y_j = \langle r(t), phi_j(t) \rangle = \int_0^T r(t)\phi_j^*(t)dt. \tag{3.4}$$

and ignoring $n(t)$ for the moment, we then recover the frequency domain data

$$Y_j = \int_0^T s(t)\phi_j^*(t)dt = \frac{1}{T}\sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} X_k \int_0^T e^{j2\pi\frac{k-j}{T}t}dt = X_j. \tag{3.5}$$

3.1.2 Implementation of OFDM system

Multiplexing N orthogonal subcarriers into an OFDM symbol can be realized by taking N-point inverse discrete Fourier transform (IDFT) on a set of N modulation

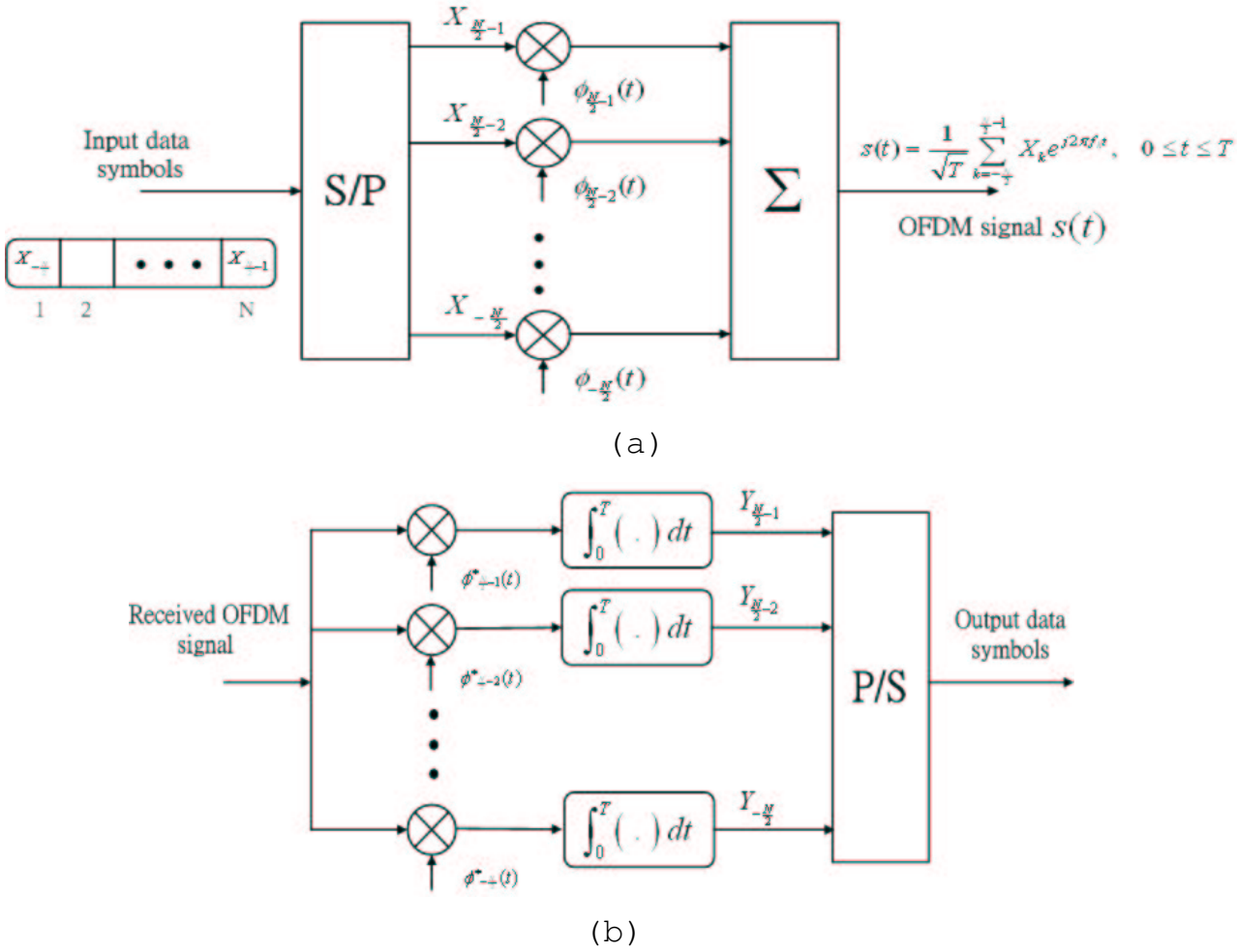


Figure 3.3: Structure of OFDM : (a)Modulator (b)Demodulator (reprinted from [11])

signals and making parallel-to-serial (P/S) and digital-to-analogous (D/A) conversions. Demultiplexing in the receiving end can be analogously realized by discrete (fast) Fourier transform (DFT) after sampling the received baseband waveform with a rate $R_s = 1/T_d = N/T$. Using the fast algorithms of the transform pair, the required complex is of order $\frac{N}{2} \cdot \log_2 N$.

Using the discrete-time signal model $r[k] = s[k] + n[k]$ and neglecting the noise

process, $n[n] = 0$, we have

$$s[n] = s(t)|_{t=nT_d} = \begin{cases} \frac{1}{\sqrt{N}} \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} X_k e^{j2\pi \frac{k}{N} n} & 0 \leqslant n \leqslant N-1 \\ 0 & \text{otherwise} \end{cases} = \mathcal{IFFT}\{X_k\} \tag{3.6}$$

$$Y_j = \mathcal{FFT}\{s[n]\} = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} s[n] e^{-j2\pi \frac{j}{N} n} = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} X_k \delta[k-j] = X_j \tag{3.7}$$

3.2 Guard time and cyclic extension

Guard interval is introduced, as mentioned before, to eliminate ISI and oftentimes its duration is about two to four times the root-mean-squared channel delay spread. Inserting a silent (virtual subcarriers) guard time between two OFDM symbols results in inter-carrier interference (ICI), induces crosstalks amongst different subcarriers, and destroys the subcarrier orthogonality. The effect is illustrated in Fig. 3.4. In this example, delayed version of subcarrier 2 causes ICI when the OFDM receiver tries to demodulate subcarrier 1, because there is no integer number of cycles difference between subcarriers 1 and 2. To avoid ICI, OFDM symbols are cyclically extended as shown in Fig. 3.5. Although an OFDM receiver only receives the sum of all these component signals, the figure gives separate component signals to illustrate the ISI effect. Suppose the multipath delay is smaller than a guard interval, there are no phase transitions during the DFT (or FFT) interval. Hence, an OFDM receiver "sees" the sum of pure sine waves with different phase offsets. That is to say, the summation does not destroy the orthogonality amongst subcarriers, it only introduces a different phase shift into each subcarrier.

Let N_g be the guard interval length and denote the extended OFDM symbol (a regular OFDM symbol with its cyclic extension) by $\tilde{s}[n]$. (3.2) will become

$$\tilde{s}[n] = \begin{cases} \frac{1}{\sqrt{N}} \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} X_k e^{j2\pi \frac{k}{N}(n-N_g)} & 0 \leqslant n \leqslant N+N_g-1 \\ 0 & \text{otherwise} \end{cases} \tag{3.8}$$

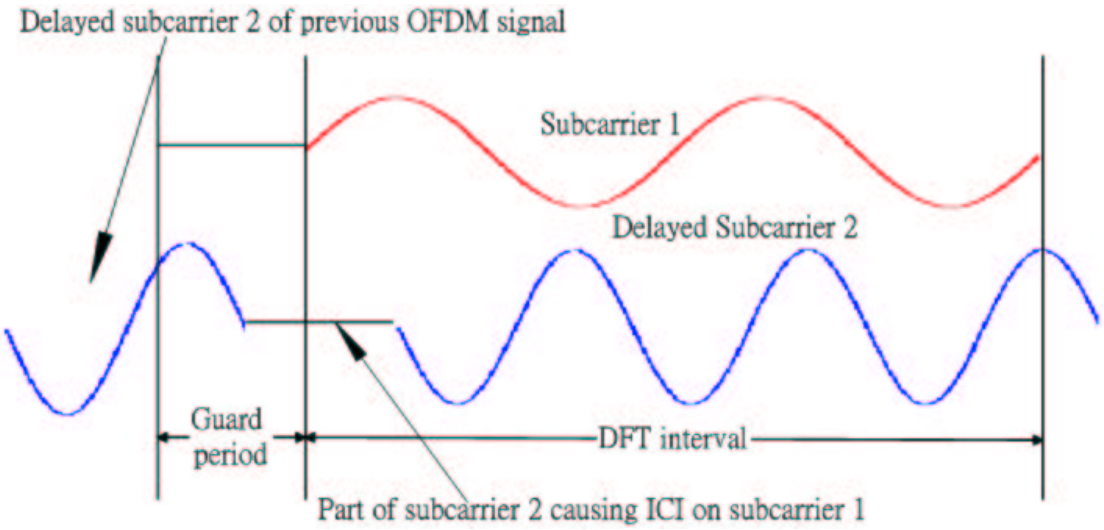


Figure 3.4: Effect of multipath with zero signal in the guard time; the delayed subcarrier 2 causes ICI on subcarrier 1 and vice versa. (reprinted from [11])

The receiver removes CP from $\tilde{r}[n]$ before performing FFT demodulation.

3.3 Windowing

In the previous sections, we have described the basic OFDM system building blocks. Since, as shown in Fig. 3.5.(a), each OFDM signal consists of a number of unfiltered subcarriers and there are discontinuities from symbol to symbol, the out-of-band spectrum decreases rather slowly, following the $sinc(\cdot)$ envelop. Fig. 3.6 plots the spectra for 16, 64, and 256 subcarriers. We notice that as the number of subcarriers increases, the corresponding out-of-band power decreases. This is because the sidelobe spacing of each subcarrier has become smaller accordingly. Basic digital filter design theory tells us that windowing is an effective means to reduce both the sidelobe hight and the out-of-band power. It also results in smoother PSD. A popular window is the raised cosine window

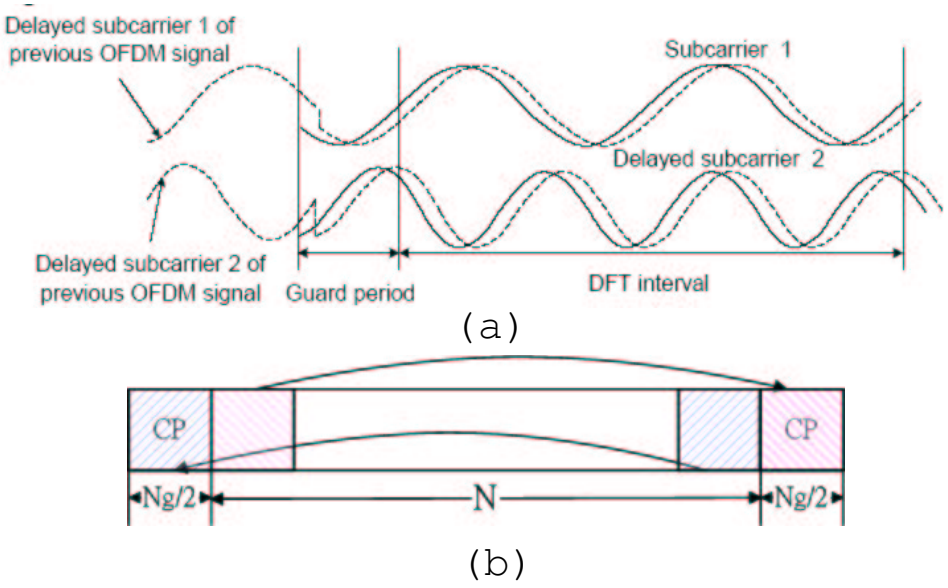


Figure 3.5: (a)The dotted curve is a delayed replica of the solid curve. (b)Cyclic prefix (CP): A copy of the last part of OFDM signal is attached to the front of itself, a copy of the first part of OFDM signal is attached to the back of itself. (reprinted from [11])

defined by

$$w_T(t) = \begin{cases} \{0.5 + 0.5\cos(\pi + t\pi/(\beta T_s)) & 0 \leqslant t \leqslant \beta T_s \\ 1.0 & \beta T_s \leqslant t \leqslant T_s \\ 0.5 + 0.5\cos((t - T_s)\pi/(\beta T_s)) & T_s \leqslant t \leqslant (1+\beta)T_s \end{cases} \tag{3.9}$$

The symbol interval T_s is shorter than the total symbol duration because we allow adjacent symbols to be partially overlapped in the roll-off region. The time structure of the OFDM signal now looks like that given in Fig. 3.7 and can be expressed as

$$s(t) = \frac{1}{\sqrt{T}} w_T(t) \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} X_k e^{j2\pi f_k(t - T_{prefix})} \qquad 0 \leqslant t \leqslant (1+\beta)T_s \tag{3.10}$$

In summary, OFDM symbols are generated as follows:

O1 N_c input (modulation) values are padded with zeros to form an N input sample vector and IFFT is performed on this vector.

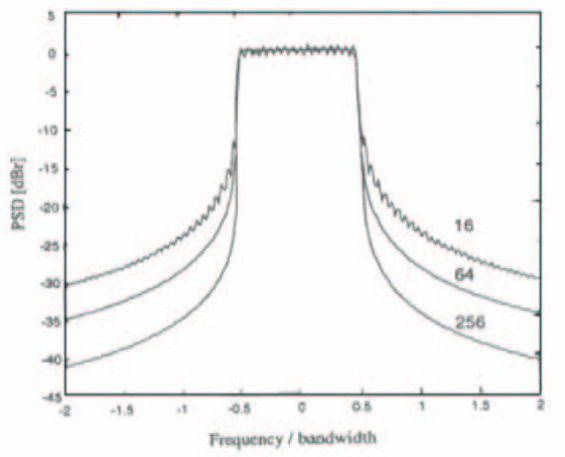


Figure 3.6: Power spectral density (PSD) without windowing for 16, 64, and 256 subcarriers. (reprinted from [11])

O2 The last T_{prefix} samples of the IFFT output are inserted at the start of the OFDM symbol, and the first $T_{postfix}$ samples are appended at the end.

O3 The OFDM symbol obtained in [O2] is multiplied by a raised cosine window w_T.

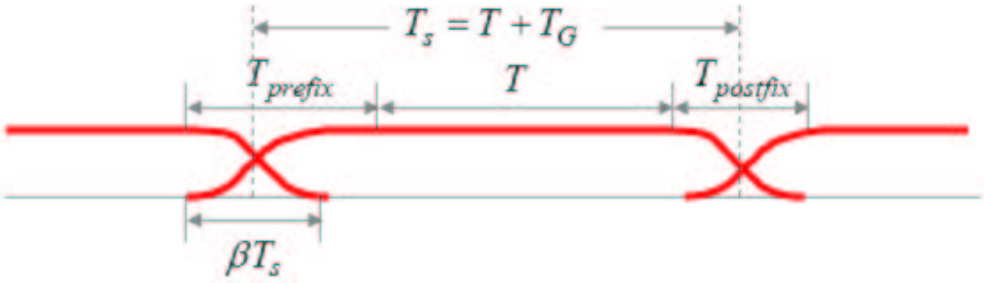


Figure 3.7: An extended OFDM symbol with cyclic extension and windowing. T_s is the symbol, T the FFT interval, T_G the guard time, T_{prefix} the preguard interval, $T_{postfix}$ the postguard interval, and β is the rolloff factor. (reprinted from [11])

Chapter 4

A Product Code Based OFDM System

This chapter considers a product code based OFDM system and the related channel estimation issue; the influence of imperfect channel estimation is considered. A model-based approach [12] for channel estimation that exploits the correlation in both frequency- and time-domain is used. The coefficients of model-based channel estimator are obtained by least-squared (LS) fitting.

To further improve the receiver performance, an iterative procedure is introduce for joint channel estimation and data detection. The tentative bit (symbol) decisions from the channel decoder output are used as pilots to re-estimate the channel response and the new channel estimates are used by the decoder to update its decoding decisions.

To begin with, we provide a short review of the model-based channel estimate.

4.1 Model-based multicarrier channel estimation

4.1.1 A mathematical model

Consider an OFDM system transmitting $\bar{N}$ OFDM symbols which is composed of $\bar{M}$ parallel channels. Denoted by $X_{\bar{m},\bar{n}}$ the symbol of the $\bar{m}$th sub-channel at the $\bar{n}$th OFDM symbol, where $0 \leq \bar{m} < \bar{M},\ 0 \leq \bar{n} < \bar{N}$. The correspondent received samples

after FFT demodulator is

$$Y_{\bar{m}\bar{n}} = H_{\bar{m}\bar{n}} X_{\bar{m}\bar{n}} + N_{\bar{m}\bar{n}}, \tag{4.1}$$

where $N_{\bar{m}\bar{n}} = N_{I,\bar{m}\bar{n}} + iN_{Q,\bar{m}\bar{n}}$ is a zero-mean complex Gaussian random variable with independent in-phase and quadrature phase components and identical variance $var(N_I) = var(N_Q) = N_0/2T \triangleq \frac{\sigma_n^2}{2}$.

The channel response (CR) can in general be modelled as an LTI system

$$h(t) = \sum_{j=1}^{L_p} h_j(t)\delta(t - \tau_j(t)), \tag{4.2}$$

where $h_j(t)$ and $\tau_j(t)$ remain constant during an extended symbol interval T_s without ICI. Hence,

$$H_{\bar{m}\bar{n}} = \sum_{j=1}^{L_p} h_j[\bar{n}] \exp\left(j2\pi\bar{m}\frac{\tau_j[\bar{n}]}{T}\right) \tag{4.3}$$

represents the channel frequency response at the $\bar{m}$th subcarrier during the $\bar{n}$th symbol interval.

Eq.(4.1) implies that, if the channel response, $H_{\bar{m}\bar{n}}$, is known, a maximum likelihood (ML) detector would make its decision based on the statistic $\hat{X}_{\bar{m}\bar{n}} = Y_{\bar{m}\bar{n}}/H_{\bar{m}\bar{n}}$. When the channel response is unknown, the receiver has to estimate the channel response $\hat{H}_{\bar{m}\bar{n}}$.

Conventional approach calls for the use of a pilot structure like that given in Fig. 4.1 to assist channel estimation. Initial channel estimate based on pilots located at $(\bar{m}, \bar{n})$ is obtained by an LS approach [4]

$$\hat{H}_{\bar{m}\bar{n},LS} = \frac{Y_{\bar{m}\bar{n}}}{X_{\bar{m}\bar{n}}} = H_{\bar{m}\bar{n}} + \frac{N_{\bar{m}\bar{n}}}{X_{\bar{m}\bar{n}}} = H_{\bar{m}\bar{n}} + V_{\bar{m}\bar{n}} \tag{4.4}$$

where $V_{\bar{m}\bar{n}}$ is the error term due to the presence of Gaussian noise whose conditional variance is given by $E[|V_{\bar{m}\bar{n}}|^2|X_{\bar{m}\bar{n}}] = 2\sigma_n^2/|X_{\bar{m}\bar{n}}|^2$.

4.1.2 Regression model based approach

The discrete channel response (CR) $H_{\bar{m}\bar{n}}$ can be viewed as a samples version of the 2-D continuous complex baseband fading process $H(f,t)$; two typical examples of

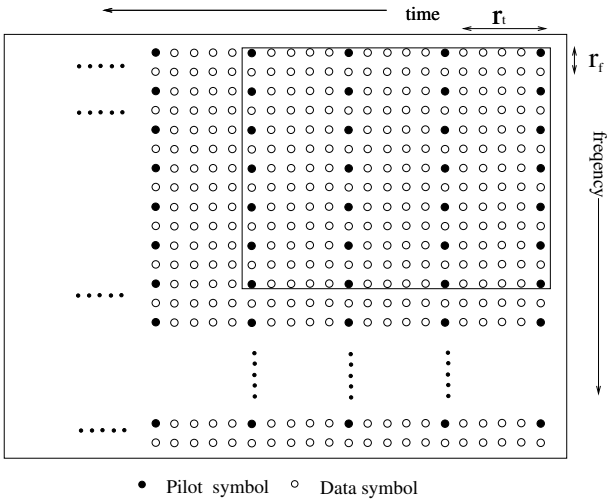


Figure 4.1: A typical pilot symbol distribution in the time-frequency plane of an OFDM signal. (reprinted from [12])

local $H_{\bar{m}\bar{n}}$, obtained by computer simulation, are shown in Fig. 4.2. We first select an operating block in the time-frequency plane in which $\bar{N}_0 \times \bar{M}_0$ pilot symbols are uniformly inserted at every r_f sub-channel and every r_t symbol; see Fig.4.1. Then the receiver models the true sampled fading process $H_{\bar{m}\bar{n}}$ in this region by a quadrature surface

$$\begin{aligned} F(\bar{m},\bar{n}) &= a\bar{m}^2 + b\bar{m}\bar{n} + c\bar{n}^2 + d\bar{m} + e\bar{n} + f \\ &= H_{\bar{m}\bar{n}} + g(\bar{m},\bar{n}) \end{aligned} \tag{4.5}$$

where $g(\bar{m},\bar{n})$ represents the modeling error. For Rician or Rayleigh fading channels, $H_{\bar{m}\bar{n}}$ is a complex Gaussian process, hence $g(\bar{m},\bar{n})$ is also complex Gaussian-distributed. The frequency-domain model of the received samples (4.1) implies that the ML estimates of the coefficients $(a,b,c,d,e,f) \triangleq \mathbf{c}^H$ are chosen such that

$$\sum_{(\bar{m},\bar{n})\in\mathcal{P}} |Y_{\bar{m}\bar{n}} - \hat{F}(\bar{m},\bar{n})X_{\bar{m}\bar{n}}|^2 = \sum_{(\bar{m},\bar{n})\in\mathcal{P}} |X_{\bar{m}\bar{n}}|^2 |\hat{H}_{\bar{m}\bar{n},LS} - \hat{F}(\bar{m},\bar{n})|^2 \tag{4.6}$$

is minimized, where $\hat{F}(\bar{m},\bar{n})$ is the estimated quadrature surface. The set $\mathcal{P}$ in (4.6)

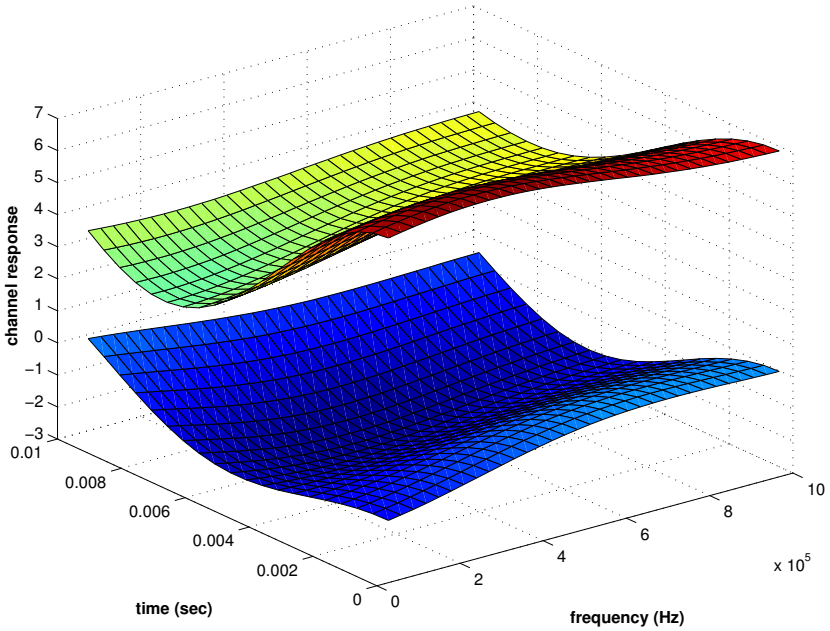


Figure 4.2: Two typical OFDM channel responses. They are plotted in the same figure for the convenience of comparison. The vertical coordinate does not represent the absolute magnitude of each CR surface. (reprinted from [12])

contains the pilot locations in the operating block (region)

$$\mathcal{P} = \left\{ (\bar{m}, \bar{n}) \;\middle|\; \begin{array}{l} \bar{m} = 0, r_f, 2r_f, \cdots, (\bar{M}_0 - 1) r_f \\ \bar{n} = 0, r_t, 2r_t, \cdots, (\bar{N}_0 - 1) r_t \end{array} \right\}. \tag{4.7}$$

Rewriting (4.5) as $F(\bar{m}, \bar{n}) = \mathbf{c}^H \mathbf{q}$, where $\mathbf{q}_{\bar{m}\bar{n}}^T \triangleq (\bar{m}^2, \bar{m}\bar{n}, \bar{n}^2, \bar{m}, \bar{n}, 1)$, we restate the problem of finding the ML solution of (4.6) as solving

$$\min_{\hat{\mathbf{c}}} \sum_{(\bar{m},\bar{n}) \in \mathcal{P}} |Y_{\bar{m}\bar{n}} - \hat{\mathbf{c}}_H \mathbf{q}_{\bar{m}\bar{n}} X_{\bar{m}\bar{n}}|^2 \tag{4.8}$$

4.1.3 Channel estimation procedure

The model-based approach described above leads to the following 2-step channel estimation procedure.

E1 Taking the derivative of (4.8) with respect to $\hat{\mathbf{c}}$ and invoking the definitions

$$\mathbf{Q} \triangleq \sum_{(\bar{m},\bar{n}) \in \mathcal{P}} \mathbf{q}_{\bar{m}\bar{n}} \mathbf{q}_{\bar{m}\bar{n}}^T |X_{\bar{m}\bar{n}}|^2, \quad \mathbf{P} \triangleq \mathbf{Q}^{-1} \tag{4.9}$$

$$\hat{\mathbf{b}} \triangleq \sum_{(\bar{m},\bar{n})\in\mathcal{P}} \mathbf{q}_{\bar{m}\bar{n}} X_{\bar{m}\bar{n}} Y^*_{\bar{m}\bar{n}}$$
$$= \sum_{(\bar{m},\bar{n})\in\mathcal{P}} \mathbf{q}_{\bar{m}\bar{n}} |X_{\bar{m}\bar{n}}|^2 \hat{H}^*_{\bar{m}\bar{n},LS} \quad (4.10)$$

where $X^*_{\bar{m}\bar{n}}$ is the complex conjugate of $X_{\bar{m}\bar{n}}$, we obtain the solution

$$\hat{\mathbf{c}} = \mathbf{P}\hat{\mathbf{b}} = \sum_{(\bar{m},\bar{n})\in\mathcal{P}} (\mathbf{P}\mathbf{q}_{\bar{m}\bar{n}} |X_{\bar{m}\bar{n}}|^2) \hat{H}^*_{\bar{m}\bar{n},LS} \quad (4.11)$$

E2 The CR estimate, $\hat{H}_{\bar{m}\bar{n}}$, for the position $(\bar{m},\bar{n})$ is

$$\hat{H}_{\bar{m}\bar{n}} = \hat{F}(\bar{m},\bar{n}) = \mathbf{q}^T_{\bar{m}\bar{n}} \hat{\mathbf{c}}^*$$
$$= \mathbf{q}^T_{\bar{m}\bar{n}} \sum_{(k,l)\in\mathcal{P}} (\mathbf{P}\mathbf{q}_{kl} |X_{kl}|^2) \hat{H}_{kl,LS} \quad (4.12)$$

The above algorithm can be modified to estimate the channel response of either a single-carrier system or any sub-channel of a multicarrier system. This 1-D scheme models the fading process by a single-variable regression function, e.g., $F(\bar{n}) = a\bar{n}^2 + b\bar{n} + c$. The corresponding parameters are given by $\mathbf{c}^H \triangleq (a,b,c)$, $\mathbf{q}^T_{\bar{n}} \triangleq (\bar{n}^2,\bar{n},1)$ and $\mathcal{P} = \{\bar{n}|\bar{n} = 0, r_t, \ldots, (\bar{N}_0 - 1)\, r_t\}$, respectively.

4.2 Joint channel estimation and TPC decoding

The invention (or re-invention) of the *turbo principle* by Berrou *et al.* [10] has far-reaching impacts on many fronts of science and engineering. In particular, we have seen the proliferation of the iterative joint estimation and detection method in designing various communication receivers. The block diagram shown in Fig. 4.3(b) is but one such application example.

Depending on the operating scenario and environment, an OFDM channel might has more sensitive frequency selectivity or time selectivity. In a static and small area application like indoor wireless communications, the frequency responses at adjacent subcarriers are not highly correlated and the coherent bandwidth is only a few subcarrier

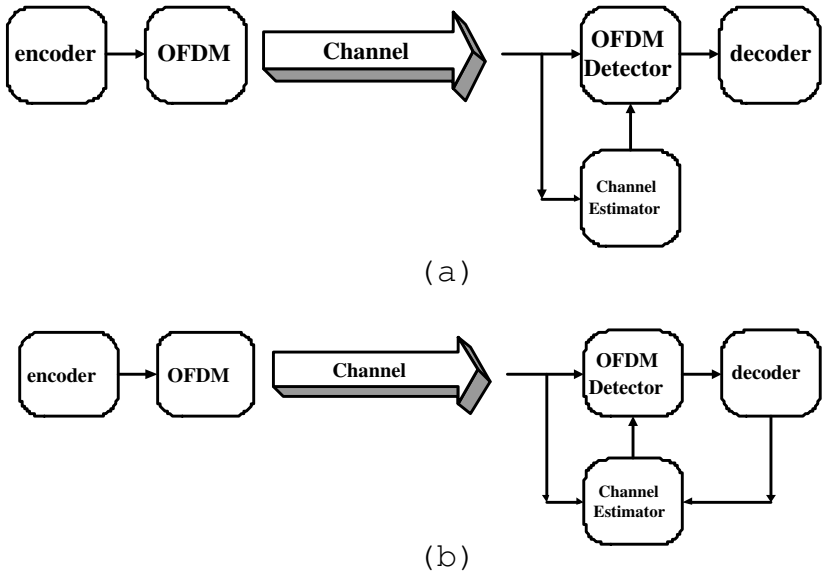


Figure 4.3: (a)A conventional OFDM system structure. (b)An OFDM system with iterative receiver structure.

spacings while the coherent time is likely to be of the order of many symbol durations. On the other hand, for cellular-like applications, we have to expect much smaller coherent times. we shall consider both cases subsequently. Take $(32, 26, 4)^2$ TPC as an example, pilots are inserted as that given in Fig. 4.4 and the size of a modelling block is $(1 \times \bar{n})$ (1-D channel estimation), namely, the pilot set $\mathcal{P}$ is separated into $\mathcal{P}_i$ $(i = 0 \sim 31)$, where

$$\begin{aligned}
\mathcal{P}_0 &= \left\{ (\bar{m}, \bar{n}) \;\middle|\; \begin{array}{l} \bar{m} = 0 \\ \bar{n} = 0, 4, 8, 13, 17, 21 \end{array} \right\} \\
\mathcal{P}_1 &= \left\{ (\bar{m}, \bar{n}) \;\middle|\; \begin{array}{l} \bar{m} = 1 \\ \bar{n} = 0, 4, 8, 13, 17, 21 \end{array} \right\} \\
&\vdots \\
\mathcal{P}_{31} &= \left\{ (\bar{m}, \bar{n}) \;\middle|\; \begin{array}{l} \bar{m} = 31 \\ \bar{n} = 0, 4, 8, 13, 17, 21 \end{array} \right\}
\end{aligned} \tag{4.13}$$

The CR estimates $\hat{H}_{\bar{m}\bar{n}}$ obtained by (4.4) and (4.12) are used to give soft input for the TPC decoder. After a few decoding iterations, the TPC decoder outputs hard decisions which are then fed back to the channel estimator to carry out another channel estimation.

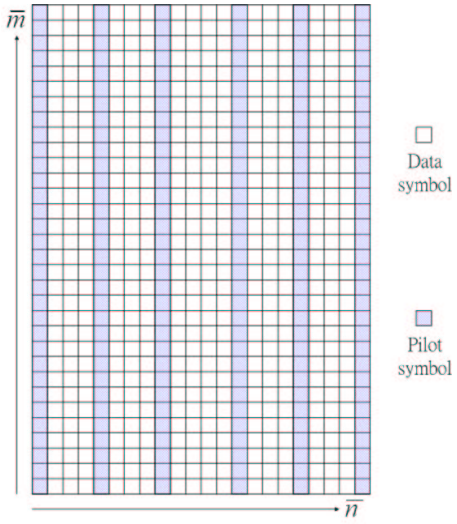


Figure 4.4: A typical pilot arrangement in the time-frequency plane.

In a new channel estimation round, the channel estimation algorithm uses both original pilots and pseudo-pilots (the decisions of the TPC decoder).

4.3 Double helical interleaver

Double helical interleaver can be inserted between the channel encoder and the OFDM modulator, permuting TPC codewords by a before OFDM modulation, to reduce the influence of long deep fade and decorrelate the channel effects on each row and column of a TPC codeword. The receiver permutes the received samples with the reverse double helical mapping and then forwards the permuted samples to the TPC decoder.

4.3.1 Structure of double helical interleaver

The double helical interleaver (DHI) can be viewed as a 2D interleaver that permutes an 2D array. For convenience, we number the entries of an $n_{\bar{m}} \times n_{\bar{n}}$ 2D array via rows and columns such that the position in the ith row of the jth column (i, j) is indexed by $j + i \times n_{\bar{n}}$. Then a DHI permute the ith entry into the jth entry by the following two

steps:

$$j' = i(n_{\bar{n}} + 1) \ (mod \ n_{\bar{n}} n_{\bar{m}}), \quad i = \{0, 1, 2, \ldots, n_{\bar{n}} n_{\bar{m}} - 1\}, \tag{4.14}$$

$$j = j'(n_{\bar{m}} + 1) \ (mod \ n_{\bar{n}} n_{\bar{m}}), \quad j' = \{0, 1, 2, \ldots, n_{\bar{n}} n_{\bar{m}} - 1\}, \tag{4.15}$$

Fig. 4.5(a) and (b) give an example of the above two DHI processing steps. Fig. 4.5(a) shows the first round of DHI process using (4.14). Entries 0 to 63 are read into the diagonal row, the nearest lower diagonal-parallel (LDP) row consists of entries 64 to 126, and the next symbol is read into upper right corner of farthest upper diagonal-parallel (UDP) row, followed by the row just below the previous LDP row and then one below the previous UDP row. The process continues, alternating between lower and upper diagonal-parallel rows, until the last entry has been filled. The second DHI step follows an order which is "complementary" to the first step. As illustrated in Fig. 4.5(b), it starts at the diagonal row, wrapped around to the nearest UDP row then to the farthest LDP row, followed by the row just above the previous UDP row, ... etc. The process continues, alternating between UDP and LDP rows, until all entries in the array are filled.

4.4 Simulation results

When the double helical interleaver is used in the TPC coded OFDM system of Fig. 4.3(b) as the channel interleaver, we have the system shown in Fig. 4.6. As mentioned above, the OFDM outputs are deinterleaved before TPC decoding commences and the decoded hard decisions are sent back to the channel estimator via the double helical interleaver for iterative channel estimation.

Numerical examples presented in this section assume that the TPC used is composed of two extended $(32, 26, 4)$ BCH codes which is the same as the extended Hamming code and the code rate is 0.5625. We use the parameter $p = 4$ and *Proakis C* five paths channel model [14] with power profile $\{0.227, 0.460, 0.688, 0.460, 0.227\}$ at delays equals

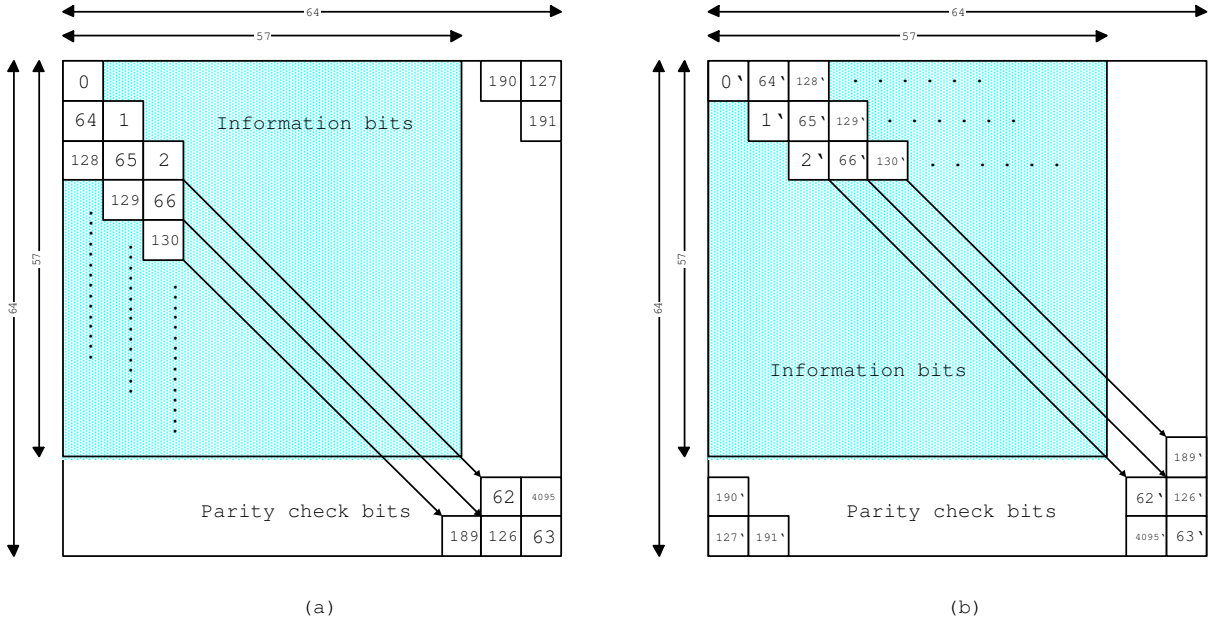


Figure 4.5: Double helical interleaving of $(64, 57, 4) \times (64, 57, 4)$ TPC

to $\{0, T, 2T, 3T, 4T\}$. Jakes' model [13] is used to generate independent fading processes associated with each path.

The effectiveness of helical interleaving on the BER performance is demonstrated in Fig. 4.7, assuming $f_m T = 0.0001$, where f_m is the maximum doppler shift and T is the sample period. The dashed-curves represent the performance of the system without channel interleaver and the solid curves represent that of the system with the double helical interleaver. The inclusion of a helical interleaver gives a 3 dB performance gain at BER=10^{-5}. The slope of the performance curve is sharper because the interleaver de-correlates the time and frequency correlations of the slow fading process; see Fig. 4.8. Fig. 4.8 plots the time-frequency channel response without (a) and with (b) the DHI. It is clear that the channel response associated with different BCH codewords and code bits are less correlated after interleaving.

In general, a higher modelling order provides more accurate channel description at the cost of larger noise-induced variance as more parameters are involved in the estimation process. At high SNRs the perturbation caused by noise is negligible hence

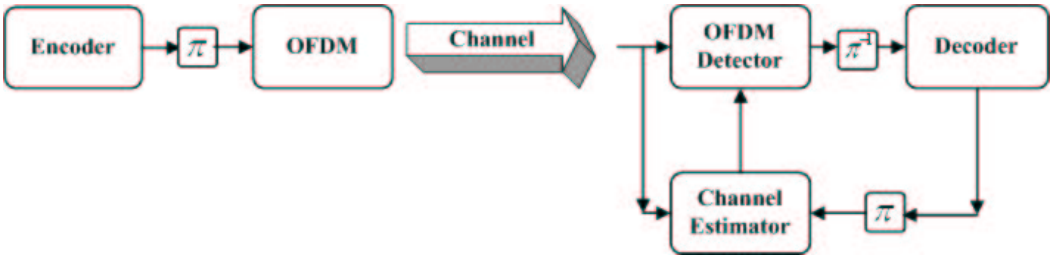


Figure 4.6: A block diagram of a double helical interleaved OFDM system with iterative joint channel estimation and TPC decoding.

higher-order estimates are preferred if complexity is not of high concern. When SNR is low, however, one would rather use a lower-order estimate as the received vector(s) is corrupted. The choice of the channel model order also depends on the channel dynamic and the modelling block size. For slow fading channels or small modelling blocks, a lower-order estimate is a better choice while for other scenarios, one might consider higher-order estimates. Hence, for a particular channel condition (SNR and coherent time/bandwidth) and modelling block size, there always exists an optimal modelling order.

Fig. 4.9 plots the system performance in a fast-fading channel ($f_m T = 0.001$). The system performance for this channel is better than that in the same channel with smaller Doppler-time product $f_m T = 0.0001$. When perfect channel estimation is available, it gives more than 5 dB improvement at BER=10^{-5} without DHI and more than 4 dB improvement at BER=10^{-5} with DHI. For such a fast-fading channel, the first-order model is clearly not sufficient to characterize the channel response. A higher-order model is needed and the simulation results indicate that a 3rd-order model yields the best performance.

At high SNRs, modelling error dominates the performance of the channel estimator [15] and the performance curves shown in Fig. 4.9 suggest that the BER performance is bounded at approximately 2×10^{-5} and 5×10^{-5} for the 3rd-order and 4th-order

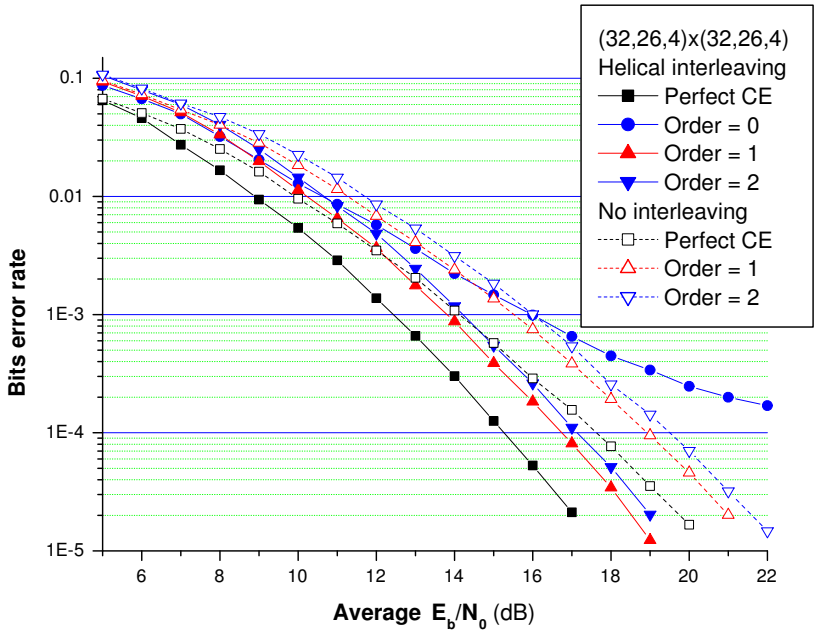


Figure 4.7: Performance of DHI-permuted system in a fading channel with $f_m T = 0.0001$.

channel estimators when there is no channel interleaving. With the DHI in place, both the performance and the corresponding bounds are improved–from 5×10^{-5} to 3×10^{-6} for the 4th-order channel estimator.

Pilot planning influences the performance of a system using pilot-assisted synchronization and channel estimation. The proposed system uses polynomials to model the true CR. Such a model tends to become less reliable when extrapolating beyond the pilot distribution boundary. For a given time-frequency block (or a time or frequency interval) within which pilots are inserted, the estimated CR tends to be less reliable in places close to the block or interval boundary. One way to remedy such a shortcoming is to place pilots in the modelling block boundary. The following equations, (4.16) and (4.17), define two pilot patterns $\mathcal{P}_a(i)$ and $\mathcal{P}_b(i)$. The former pattern has pilots on its

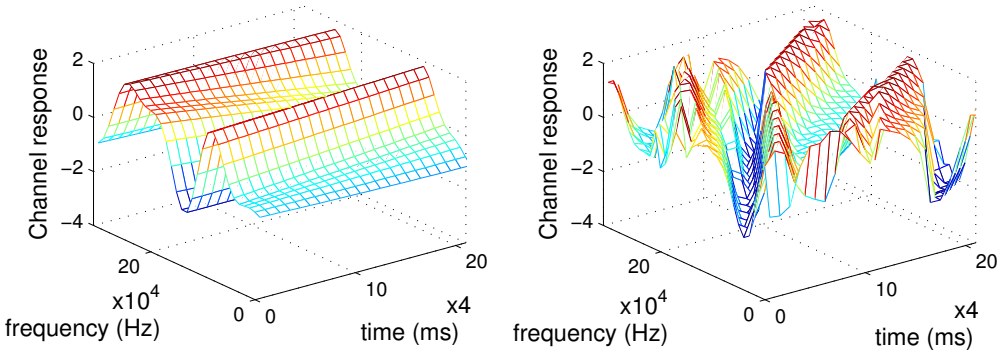


Figure 4.8: (a) Time-frequency channel response with $f_m T = 0.0001$; (b) Channel response with DHI permutation.

modelling block edges but not the latter.

$$\mathcal{P}_a(i) = \left\{ (\bar{m}, \bar{n}) \;\middle|\; \begin{array}{l} \bar{m} = i \\ \bar{n} = 0, 4, 8, 13, 17, 21 \end{array} \right\}. \quad i = 0, 1, \ldots, 31 \tag{4.16}$$

$$\mathcal{P}_b(i) = \left\{ (\bar{m}, \bar{n}) \;\middle|\; \begin{array}{l} \bar{m} = i \\ \bar{n} = 2, 5, 9, 13, 16, 19 \end{array} \right\}. \quad i = 0, 1, \ldots, 31 \tag{4.17}$$

Fig. 4.11 compares the BER performance of the above two pilot patterns. The former pilot pattern results in 0.5 and 4 dB gains at BER=10^{-4} when 3rd-order and 4th-order models are used. To validate our claim that pilot pattern does affect the system performance and to examine the relation between model order and channel dynamic, we plots CRs of the two pilot patterns of the 3rd-order channel estimate (i.e., the one using a degree-3 polynomial channel model) and the true CR (marked by solid stars) in Fig. 4.12. Fig. 4.13 is similar to Fig. 4.12 except that 4th-order model is used. The square and cross markers on both figures represent CR estimates using two different pilot patterns. It can be seen that high modelling order tends to incur larger estimation errors at positions close to the edge. Using a pilot pattern with pilots in the boundary positions does help reducing the estimation error.

Finally, in Fig. 4.14 we plot the BER performance when the receiver employs joint iterative channel estimation and TPC decoding. A 3rd-order channel estimator is used

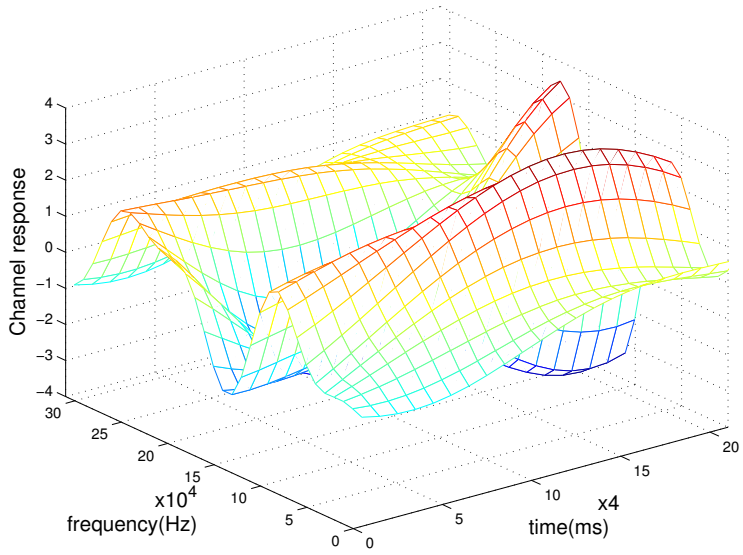


Figure 4.9: Time-frequency response of the *Proakis C* five-path fading channel with $f_m T = 0.001$.

and both two- and four-iteration detection are considered. The performance gain with respect to the no-iteration receiver is about 1.0 dB at BER = 10^{-4} when two-iteration is used and is increased by another 0.3 dB with a four-iteration receiver. As expected, iterative channel estimation does enhances the system performance, bringing about performance closer to that of the ideal receiver with perfect channel estimate.

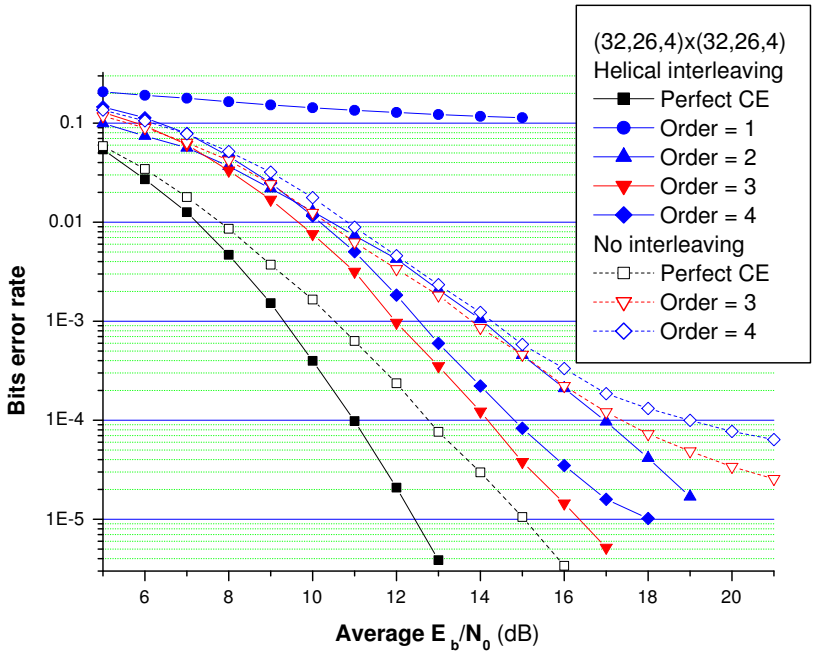


Figure 4.10: BER Performance comparison for the channel shown in the above figure.

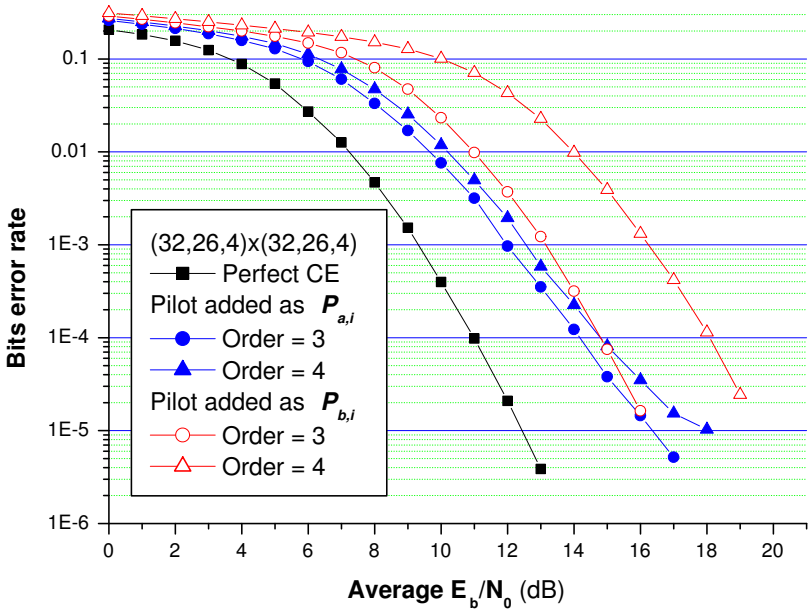


Figure 4.11: Performance curve of different pilot insert position in $f_m T = 0.001$ fading channel.

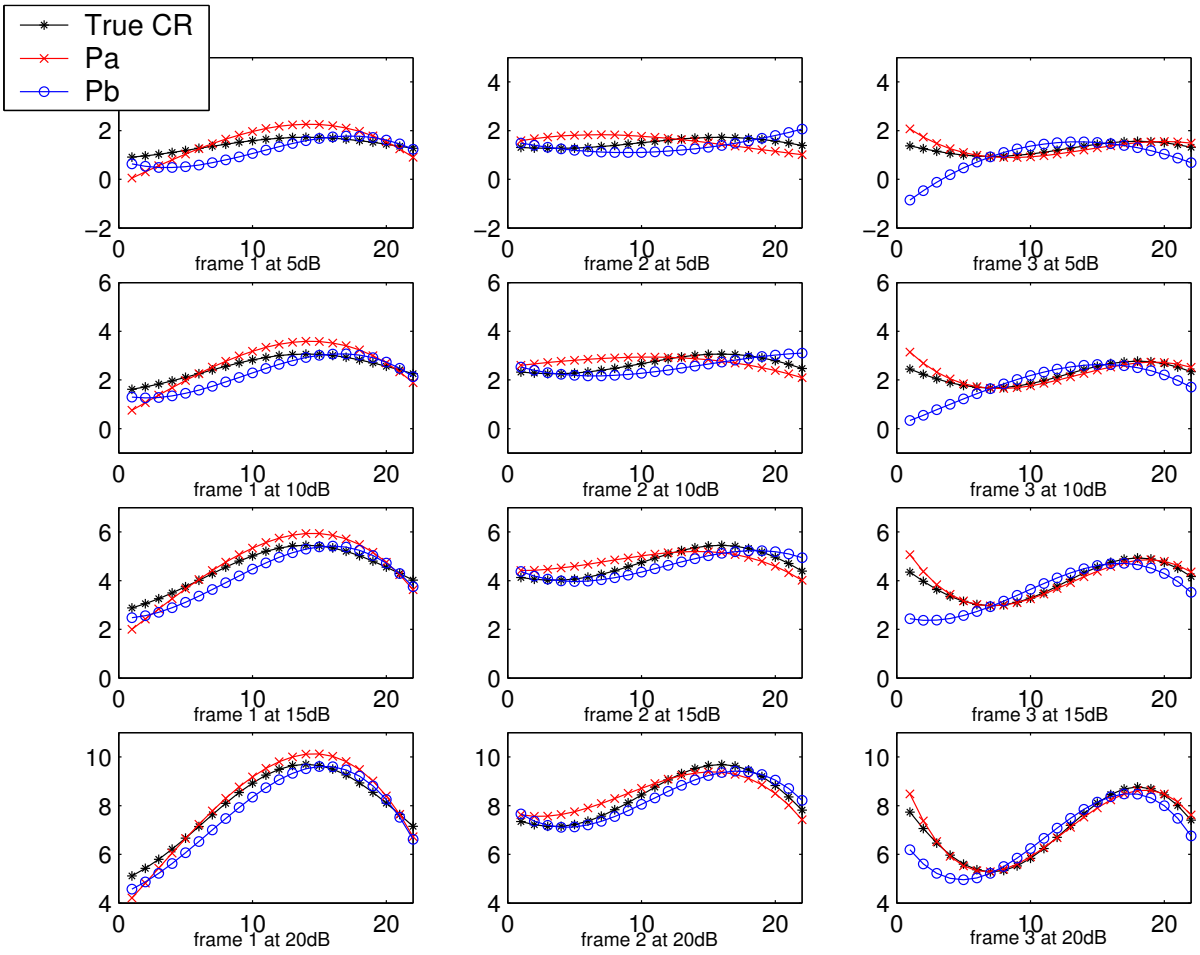


Figure 4.12: Estimated CR using a third-order channel model using pilot patterns (4.16) and (4.17); the true CR (solid star markers) is included for comparison; $f_m T = 0.001$.

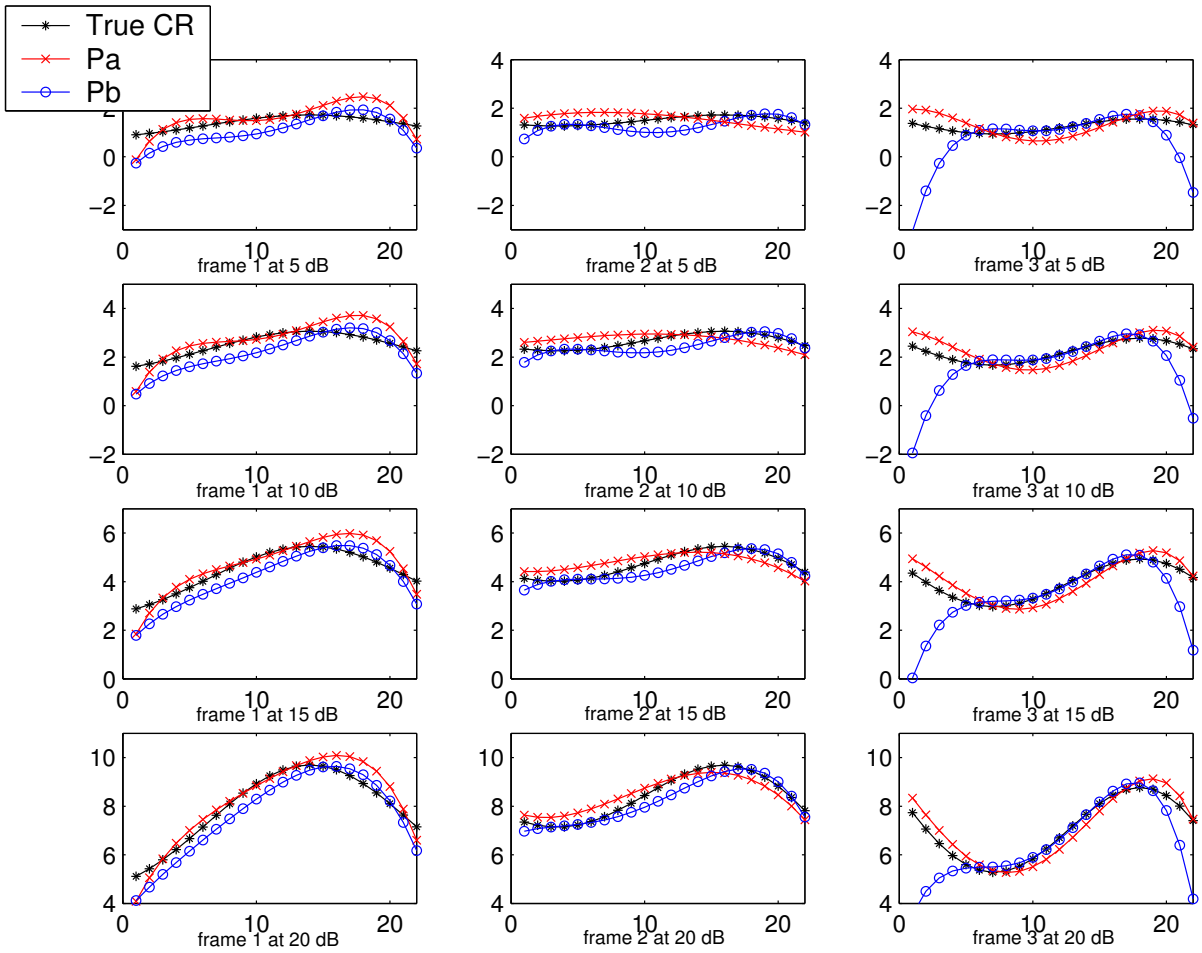


Figure 4.13: Estimated CR using a fourth-order channel model using pilot patterns (4.16) and (4.17). The true CR is marked by solid stars; $f_m T = 0.001$.

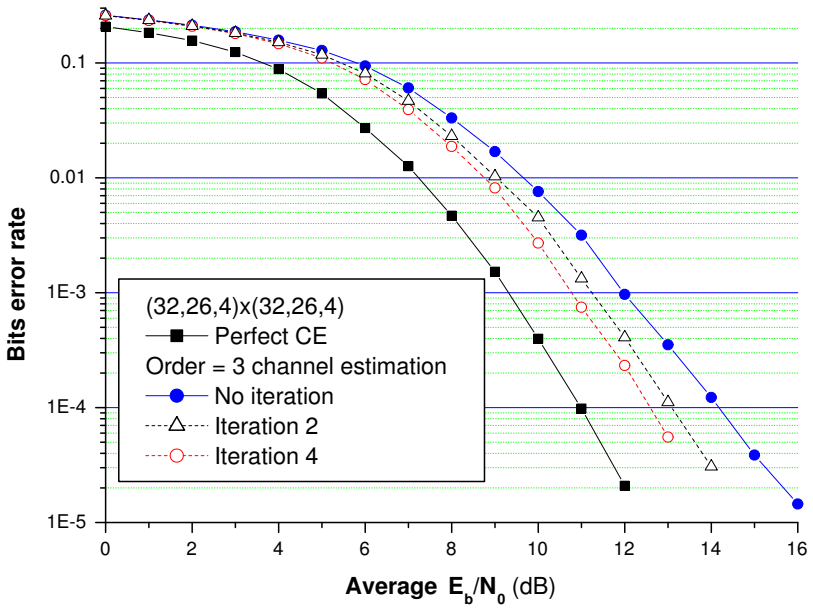


Figure 4.14: BER performance curves of the receiver with iterative joint channel estimation and TPC decoding in a fading channel with $f_m T = 0.001$.

Chapter 5

Parallel Concatenated Product Codes

We extend the concept of Pyndiah's block turbo codes and propose a new class of codes based on product codes called parallel concatenated product codes whose structure is similar to that of turbo codes [9], replacing the constituent convolutional codes in a turbo code by product codes. This new class of codes provides more flexible choices of code rates and component codes. When simple systematic product codes are used as component codes, the corresponding decoding complexity remains relatively low and the achievable performance outperforms that of a turbo product code with comparable rate at high SNR.

The interleaver we use is the Fibonacci interleaver. Besides its simplicity in implementation, we prove that such an interleaver guarantees that the resulting PCPC has a minimum distance larger than that of its constituent product codes if a proper code length is selected.

5.1 Encoder

The structure of parallel concatenated product codes (PCPC) is shown in Fig. 5.1 in which "TPC encoder" is the encoder of a product code given in Fig. 1. A codeword

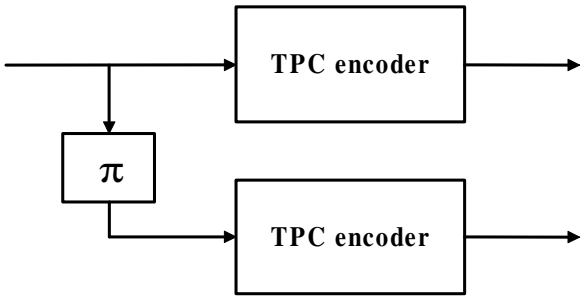


Figure 5.1: The structure of the PCPC encoder.

consists of the product codeword of the upper branch and the parity part of the lower branch output. The only interleaver considered here is the Fibonacci interleaver.

5.2 Decoder

The decoder consists of four a posteriori probability (APP) component decoders, each is responsible for decoding a component block code. Fig. 5.2 depicts the structure of a component decoder which is similar to that presented in [8]. Denote by $[\boldsymbol{R}]$ the receiving matrix and by $[\boldsymbol{W}_t(m)]$ the output extrinsic information matrix of the tth APP decoder at the end of the mth APP decoding round. The a priori information matrix given at input of the tth APP decoder is

$$\boldsymbol{W}_{sum}(m) = \sum_{l \neq t} \boldsymbol{W}_l(m). \tag{5.1}$$

The message passing operations among these four APP decoders is shown in Fig. 5.3, where $\boldsymbol{I}$ and $\boldsymbol{P}$ are extrinsic information corresponding to information bits and parity check bits, respectively. A complete iteration follows the APP decoding schedule: $\boldsymbol{APP}_0 \rightarrow \boldsymbol{APP}_1 \rightarrow \boldsymbol{APP}_2 \rightarrow \boldsymbol{APP}_3$. Each APP decoder also sends related extrinsic information to the other two decoders that do not sit next to it in the schedule. Decoding of a component TPC usually converges after 8 APP decoding rounds but a PCPC may takes an average of 16 APP decoding rounds to converge.

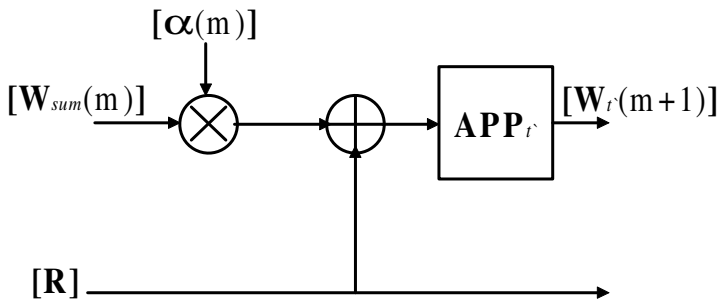


Figure 5.2: Block diagram of elementary APP decoder $\boldsymbol{APP}_{t`}$.

Similar to [8], the choice of the *weighting factor* α and *reliability factor* β is very critical in determining the performance. The sets of α and β we used for different decoding rounds are given by

$$\alpha(m) = [0.0, 0.1, 0.2, 0.3, 0.4, 0.6, 0.8, 1.0]. \tag{5.2}$$

$$\beta(m) = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.8, 1.0]. \tag{5.3}$$

and

$$\begin{aligned}\alpha(m) = [&0.0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35,\\ &0.4, 0.45, 0.5, 0.55, 0.6, 0.7, 0.8, 1.0].\end{aligned} \tag{5.4}$$

$$\begin{aligned}\beta(m) = [&0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45,\\ &0.5, 0.55, 0.6, 0.65, 0.7, 0.8, 0.9, 1.0].\end{aligned} \tag{5.5}$$

5.3 Fibonacci interleaver

The position of each bit in the $(k_1 \times k_2)$ information array is represented by $(\bar{i}, \bar{j})$, where $0 \leq \bar{i} < k_1$ and $0 \leq \bar{j} < k_2$. Using the definition $|W|_Z = W \pmod{Z}$, the Fibonacci interleaver is characterized by the permutation rule

$$(\bar{i}, \bar{j}) \rightarrow (|\bar{i} + \bar{j}|_{k_1}, ||\bar{i} + \bar{j}|_{k_1} + \bar{j}|_{k_2}). \tag{5.6}$$

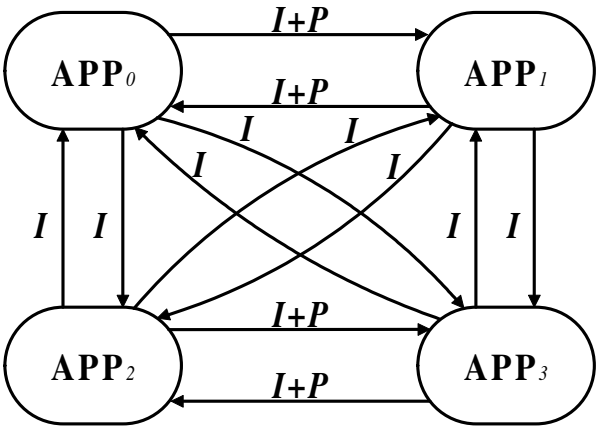


Figure 5.3: An flow chart showing the message-passing of various component APP decoders and related decoding schedule.

For a product code with extended Hamming codes as component codes, a minimum-weight codeword has nonzero entries at the same locations of some four nonzero-weight rows and columns within the information array, i.e., the parity part of the product codeword has weight 0. The basic requirement of the interleaver used in a PCPC is to make sure the interleaved information array will not contain such a 4×4 all-1 subarray. The Fibonacci interleaver does possess such a desired property. In particular, one can show that

Lemma 1. *A PCPC using* $(n_1, k_1, 4) \times (n_2, k_2, 4)$ *product codes as constituent codes has a minimum Hamming distance greater than 16 if* k_1 *or* k_2 *is not a multiple of* 4.

Proof. Since each component block code of a constituent product code has minimum distance 4, we use a 4×4 matrix called minimum-weight error event array to represent the nonzero positions of a minimum weight product codeword. As it is possible that all entries of this matrix lie within the information array of a product codeword, we want the interleaver to guarantee that the permuted information array will not produce an all-zero parity part.

Without loss of generosity, let the nonzero positions be at $(\bar{j}, \bar{j} + g_1, \bar{j} + g_2, \bar{j} + g_3)$

for four columns and $(\bar{i}, \bar{i}+h_1, \bar{i}+h_2, \bar{i}+h_3)$ for four rows, where $0 < h_1 < h_2 < h_3 < k_1$ and $0 < g_1 < g_2 < g_3 < k_2$. We have the error-position array $[\gamma_{w,z}]_{4\times 4}$ denoted by

$$\begin{bmatrix} (\bar{i},\bar{j}) & \cdots & \cdots & (\bar{i},\bar{j}+g_3) \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ (\bar{i}+h_3,\bar{j}) & \cdots & \cdots & (\bar{i}+h_3,\bar{j}+g_3) \end{bmatrix} \tag{5.7}$$

where $1 \le w, z \le 4$, $\gamma_{w,z} = (\bar{i}+u_w, \bar{j}+v_z)$, $\mathbf{u} = (u_1, u_2, u_3, u_4) = (0, h_1, h_2, h_3)$ and $\mathbf{v} = (v_1, v_2, v_3, v_4) = (0, g_1, g_2, g_3)$.

After Fibonacci interleaving, the positions will be permuted to

$$\tilde{\gamma}_{w,z} = (|\bar{i}+u_w+\bar{j}+v_z|_{k_1}, ||\bar{i}+u_w+\bar{j}+v_z|_{k_1}+\bar{j}+v_z|_{k_2}).$$

which can be decomposed as

$$\begin{aligned} \tilde{\gamma}_{w,z} &= (|\bar{i}+\bar{j}|_{k_1}, ||\bar{i}+\bar{j}|_{k_1}+\bar{j}|_{k_2}) \\ &\quad \bigoplus_{k_1\times k_2} (|u_w+v_z|_{k_1}, ||u_w+v_z|_{k_1}+v_z|_{k_2}), \\ &= \dot{\gamma}_{w,z} \bigoplus_{k_1\times k_2} \ddot{\gamma}_{w,z}. \end{aligned}$$

where $\bigoplus_{k_1\times k_2}$ represents the operation that takes module k_1 and k_2 on the first and the second entries, respectively. Because all entries of the array $[\dot{\gamma}_{w,z}]_{4\times 4}$ are all the same, $[\tilde{\gamma}_{w,z}]_{4\times 4}$ does not form a matrix with all entries forming a 4 by 4 position array only if $[\ddot{\gamma}_{w,z}]_{4\times 4}$ does not. Therefore, we only need to consider an error-position array

$$\begin{aligned} [\ddot{\gamma}_{w,z}]_{4\times 4} &= [|u_w+v_z|_{k_1}, ||u_w+v_z|_{k_1}+v_z|_{k_2}]_{4\times 4}\,, \\ &= [\hat{\gamma}_{w,z}, \check{\gamma}_{w,z}]_{4\times 4}\,. \end{aligned}$$

where $\hat{\gamma}_{w,z}$ and $\check{\gamma}_{w,z}$ represents the rows and columns of this position array, respectively. As we know, the array $[\ddot{\gamma}_{w,z}]_{4\times 4}$ will form a 4×4 position array without generating parity bits only if there are four different kinds of values in rows or columns. Therefore, we

firstly consider an array of the form

$$[\hat{\gamma}_{w,z}]_{4\times 4} = \begin{bmatrix} |0|_{k_1} & |g_1|_{k_1} & |g_2|_{k_1} & |g_3|_{k_1} \\ |h_1|_{k_1} & |h_1+g_1|_{k_1} & |h_1+g_2|_{k_1} & |h_1+g_3|_{k_1} \\ |h_2|_{k_1} & |h_2+g_1|_{k_1} & |h_2+g_2|_{k_1} & |h_2+g_3|_{k_1} \\ |h_3|_{k_1} & |h_3+g_1|_{k_1} & |h_3+g_2|_{k_1} & |h_3+g_3|_{k_1} \end{bmatrix}.$$

If these 16 positions are permuted to the same four rows $0, g_1, g_2, g_3,$ then h_1, h_2, h_3 should be moved to g_1, g_2, g_3. If h_1 is moved to g_2 or g_3, either h_2 or h_3 should be mapped to g_1, which is obviously a contradiction. Therefore, we have $h_1 = g_1$. If h_2 is mapped to g_3 then h_3 should be mapped to g_2, which is not possible, therefore, $h_2 = g_2$ and $h_3 = g_3$ and the resulting array becomes

$$[\hat{\gamma}_{w,z}]_{4\times 4} = \begin{bmatrix} |0|_{k_1} & |g_1|_{k_1} & |g_2|_{k_1} & |g_3|_{k_1} \\ |g_1|_{k_1} & |2g_1|_{k_1} & |g_1+g_2|_{k_1} & |g_1+g_3|_{k_1} \\ |g_2|_{k_1} & |g_2+g_1|_{k_1} & |2g_2|_{k_1} & |g_2+g_3|_{k_1} \\ |g_3|_{k_1} & |g_3+g_1|_{k_1} & |g_3+g_2|_{k_1} & |2g_3|_{k_1} \end{bmatrix}.$$

If $\hat{\gamma}_{w,w} \neq 0$ for $w = 2, 3, 4$, then there are three entries $\hat{\gamma}_{w,z} = 0,\ w \neq z,\ 0 < w, z \leq 4$, which again is a contradiction. Therefore, at least one of $\hat{\gamma}_{w,w}$ is 0 for $w = 2, 3, 4$. We now proceed to discuss these three cases.

1. $|2g_1|_{k_1} = 0 :\ g_1 = \frac{k_1}{2}$ implies $g_1 + g_2 \neq 0$, $g_1 + g_3 \neq 0$. Thus $g_2 + g_3$ must be equal to 0 but then we have $k_1 > g_3 > g_2 > g_1 = \frac{k_1}{2}$, a contradiction.

2. $|g_2|_{k_1} = 0 :\ g_2 = \frac{k_1}{2}$ implies $g_2 + g_3 \neq 0$, $g_1 + g_2 \neq 0$ and $g_1 + g_3 = 0$. Therefore $\mathbf{v} = (0, g_1, \frac{k_1}{2}, k_1 - g_1)$ and $[\hat{\gamma}_{w,z}]_{4\times 4}$ becomes

$$\begin{bmatrix} 0 & g_1 & \frac{k_1}{2} & k_1 - g_1 \\ g_1 & 2g_1 & g_1 + \frac{k_1}{2} & 0 \\ \frac{k_1}{2} & g_1 + \frac{k_1}{2} & 0 & \frac{k_1}{2} - g_1 \\ k_1 - g_1 & 0 & \frac{k_1}{2} - g_1 & -2g_1 \end{bmatrix}. \tag{5.8}$$

   Because there should be four entries equal to "$\frac{k_1}{2}$", $2g_1$ and $-2g_1$ are equal to "$\frac{k_1}{2}$" and $g_1 = \frac{k_1}{4}$, hence $\mathbf{v} = (0, \frac{k_1}{4}, \frac{k_1}{2}, \frac{3k_1}{4})$.

3. $|2g_3|_{k_1} = 0 :\ g_3 = \frac{k_1}{2}$ implies $g_1 + g_3 \neq 0$, $g_2 + g_3 \neq 0$, which forces $g_1 + g_2$ to be equal to 0 but then we have the contradictory result, $0 < g_1 < g_2 < g_3 = \frac{k_1}{2}$.

Next let us consider the sixteen values of the columns error position array $[\check{\gamma}_{w,z}]_{4\times 4} = [||v_w + v_z|_{k_1} + v_z|_{k_2}]_{4\times 4}$. By substituting $\mathbf{v} = (0, \frac{k_1}{4}, \frac{k_1}{2}, \frac{3k_1}{4})$ into the array, we have

$$\begin{bmatrix} 0 & |\frac{k_1}{2}|_{k_2} & |k_1|_{k_2} & |\frac{3k_1}{2}|_{k_2} \\ |\frac{k_1}{4}|_{k_2} & |\frac{3k_1}{4}|_{k_2} & |\frac{5k_1}{4}|_{k_2} & |\frac{3k_1}{4}|_{k_2} \\ |\frac{k_1}{2}|_{k_2} & |k_1|_{k_2} & |\frac{k_1}{2}|_{k_2} & |k_1|_{k_2} \\ |\frac{3k_1}{4}|_{k_2} & |\frac{k_1}{4}|_{k_2} & |\frac{3k_1}{4}|_{k_2} & |\frac{5k_1}{4}|_{k_2} \end{bmatrix}. \tag{5.9}$$

There are four "$|\frac{3k_1}{4}|_{k_2}$", three "$|k_1|_{k_2}$", three "$|\frac{k_1}{2}|_{k_2}$", two "$|\frac{1k_1}{4}|_{k_2}$", two "$|\frac{5k_1}{4}|_{k_2}$", one "$|\frac{6k_1}{4}|_{k_2}$" and one "0" in these entries. Because only four kinds of values are allowed in these entries and $|\frac{1k_1}{4}|_{k_2}$, $|\frac{5k_1}{4}|_{k_2}$ and $|\frac{6k_1}{4}|_{k_2}$ can not be "0", the remaining situations are

1. $|k_1|_{k_2} = 0 :\ k_1 = l \times k_2, l \in \mathbf{Z}.\ 0, |\frac{lk_2}{4}|_{k_2}, |\frac{lk_2}{2}|_{k_2}, |\frac{3lk_2}{4}|_{k_2}$ should be different and it implies l is odd.

2. $|\frac{k_1}{2}|_{k_2} = 0 :\ k_1 = 2l \times k_2, l \in \mathbf{Z}$. This implies $|\frac{k_1}{4}|_{k_2} = |\frac{3k_1}{4}|_{k_2}$ and contradicts.

However, either k_1 or k_2 is not a multiple of 4 and there is no 4×4 array resulting in the PCPC codeword weight of 16. □

Employing an analogous argument we can also prove

Lemma 2. *A PCPC using $(n_1, k_1, 4) \times (n_2, k_2, 4)$ product codes as constituent codes cannot have a minimum Hamming distance equals to 17, 18 or 19.*

Lemmas 1 and *2* implies

Theorem 1. *A PCPC using $(n_1, k_1, 4) \times (n_2, k_2, 4)$ product codes as constituent codes has a minimum Hamming distance on less than 20 if k_1 or k_2 is not a multiple of* 4.

5.4 Simulation results

We give computer-simulated performance of two PCPCs. The first PCPC is based on $(32, 26, 4)$ extended Hamming codes and is denoted by $(32, 26, 4)^4$. The second one is build on the product code using the $(64, 57, 4)$ and $(16, 11, 4)$ extended Hamming code as component code, hence it is denoted by $(64, 57, 4)^2 \times (16, 11, 4)^2$. These two codes have code rates, $26^2/[2(32^2) - 26^2] = .493$, $(57 \times 11)/[2(64 \times 16) - (57 \times 11)] = .441$, which are comparable to that of the $(16, 11, 4)^2$ product code which is based on the $(16, 11, 4)$ extended Hamming code.

For fair comparison, we define a decoding round as one-pass decoding of a product code $\boldsymbol{APP}_i \rightarrow \boldsymbol{APP}_{i+1}$. Thus, a decoding iteration of PCPC needs two consecutive decoding rounds. The performance curves shown in Fig. 5.4 include those obtained by 2 and 4 decoding iterations of the $(32, 26, 4)^4$ PCPC and that by 4 decoding rounds of the $(16, 11, 4)^2$ product code. The performance of the $(32, 26, 4)^4$ PCPC with 4 decoding iterations is superior to that with 2 decoding iterations by 0.5 dB at BER=10^{-5}. The $(32, 26, 4)^4$ PCPC with 4 decoding iterations also outperforms the $(16, 11, 4)^2$ TPC with 4 decoding iterations by 0.5 dB at the BER=10^{-5}. The performance gain improves if we are interested in lower BER performance. For $(16, 11, 4)^2$ TPC, performance improvement becomes negligible when the number of decoding iterations is greater than 4. The PCPCs, however, require longer convergence times. We also find that with the same decoding iterations, the $(64, 57, 4)^2 \times (16, 11, 4)^2$ PCPC outperforms the $(32, 26, 4)^4$ PCPC by 0.2 dB.

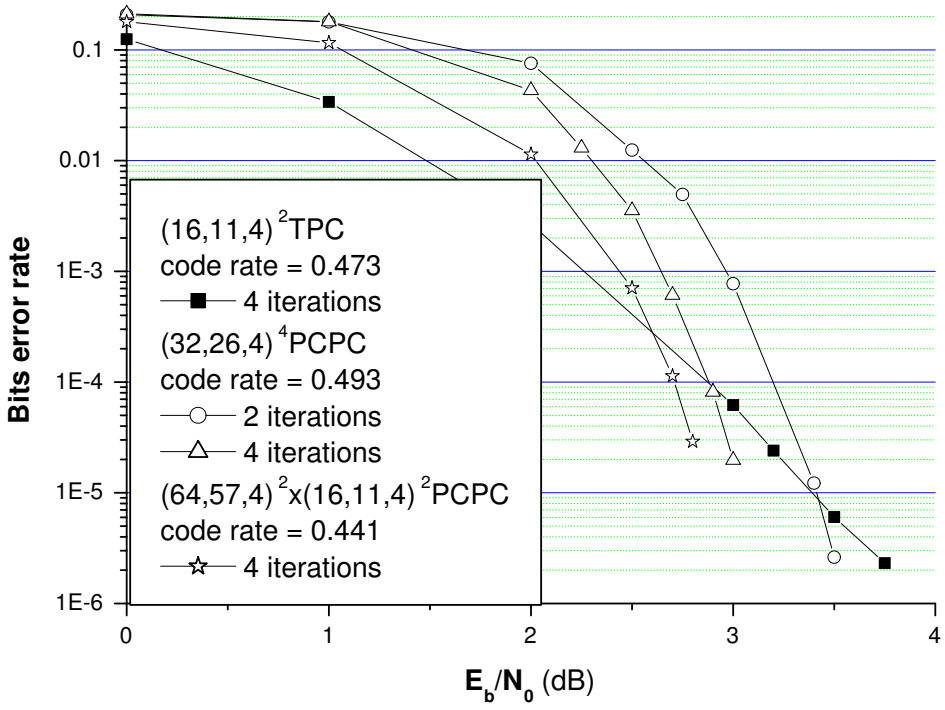


Figure 5.4: Bit error rate performance of two PCPCs and a TPC.

Chapter 6

Conclusion

We study the architecture and performance of product-coded OFDM systems over frequency selective fading channels. Joint channel estimation and TPC decoding algorithms are proposed to enhance the receiver performance. Model-based approach is used for channel estimation and the issues of pilot distribution and modelling order are addressed. We find that it is important to place pilots on the edge of each modelling block and the modelling order should match both the channel dynamic (coherent time/bandwidth) and the modelling block size.

To further decorrelate the frequency- and time-selective fading effect, the inclusion of an additional interleaver–the double helical interleaver–has proved to be very fruitful, bringing about performance gain of 3-5 dB.

We also propose a new product code based system–the PCPC systems–one that replace the recursive systematic convolutional codes in a turbo encoder by systematic product codes. Such a code offers rate flexibility and code extension capability. With proper selections on the code parameters and interleaver, the PCPC system can offer an outstanding performance with moderate decoding complexity.

The PCPC code is suitable for ARQ system as well though we do not have time to look into the associated design and performance. Although we have used an inner loop/outer loop decoding schedule, an optimal schedule remains unknown and further

studies are called for. The basic minimum distance low bound of a PCPC using the Fibonacci interleaver is established but the exact minimum distance has not been found and whether the Fibonacci interleaver is the best choice remains unanswered.

Moreover, we believe that the structure of the interleaver and the product code configurations have a very close relationship. Current product code has a configuration similar to a block interleaver, which does not match the channel's time-frequency response. A near optimal design should try to match the above system design parameters (interleaver structure, code and OFDM arrangements) with the channel response.

Bibliography

[1] H. Burton, E. Weldon, Jr, "Cyclic product codes," *IEEE Trans. Inform. Theory,* vol. 11, pp. 433 -439, Jul. 1965.

[2] R. Pyndiah, A. Glavieux, A. Picart, S. Jacq, "Near optimum decoding of product codes ," in *Proc. IEEE Global Commun. Conf.*, vol. 1, pp. 339 - 343, 28 Nov.-2 Dec. 1994.

[3] S. A. Hirst, B. Honary, G. Markarian, "Fast Chase algorithm with an application in turbo decoding" *IEEE Trans. Commun.,* vol. 49, pp. 1693-1699, Oct. 2001.

[4] J.-J. van de Beek, O. Edfors, M. Sandell, and S. K. Wilson, "On channel estimation in OFDM systems," in *Proc. 45th IEEE Vehicular Technology Conf., Chicago,* IL, July 1995, pp. 815-819.

[5] Stephen B. Wicker, *Error control systems for digital communication and storage*, Prentice Hall, 1995.

[6] $\cdots$,*Concatenated Codes.* Cambridge, MA:MIT Press, 1996.

[7] D. Chase, "A class of algorithm for decoding block codes with channel measurement information," *IEEE Trans. Inform. Theory*, vol IT-18, pp.170-182, Jan. 1972.

[8] R.M. Pyndiah,"Near-optimum decoding of product codes: Block turbo codes," *IEEE Trans. Commun.*, vol. 46, pp. 1003 - 1010, Aug. 1998.

[9] C. Berrou, A. Glavieux, "Near optimum error correcting coding and decoding: Turbo code," *IEEE Trans. Commun.*, vol. 44, pp. 1261-1271, Oct. 1996.

[10] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo- codes (1)," in *IEEE Int. Conf. Communications ICC'93,* vol. 2/3, pp. 1064-1071, May 1993.

[11] Richard D.J. van Nee, Ramjee Prasad, *OFDM for wireless multimedia communications*, Artech House Publishers, 2000.

[12] M.-X. Chang and Y. T. Su, "Model-based channel estimation for OFDM signals in Rayleigh fading," *IEEE Trans. Commun.*, vol. 50, pp. 540-544, Apr. 2002.

[13] W. C. Jakes, Jr., *Microwave Mobile Communications*. New York: Wiley, 1974.

[14] J. G. Proakis, *Digital Communications*, McGraw-Hills, 4th edition, 2001.

[15] M.-X. Chang and Y. T. Su, "Performance analysis of Equalized OFDM systems in Rayleigh fading," *IEEE Trans. Wireless Commun.*, vol. 1, pp. 712-720, Oct. 2002.