Turbo Coding, Turbo Equalisation and Space-Time Coding:
EXIT-Chart Aided Near-Capacity Designs for Wireless Channels

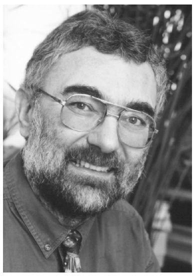
by

©L. Hanzo, T. H. Liew, B. L. Yeap, R.Y.S. Tee
School of Electronics and Computer Science,
University of Southampton, UK

We dedicate this monograph to the contributors of this field listed in the Author Index

Contents

About the Authors



Lajos Hanzo, Fellow of the Royal Academy of Engineering, graduated in Electronics in 1976 and in 1983 he was conferred a doctorate in the field of Telecommunications. In 2004 He received a DSc for his research on adaptive wireless transceivers from the University of Southampton. During his 34-year career in telecommunications he has held various research and academic posts in Hungary, Germany and the UK. Since 1986 he has been with the School of Electronics and Computer Science, University of Southampton, UK and has been a consultant to Multiple Access Communications Ltd., UK. He currently holds the established Chair of Telecommunications.

He co-authored 19 Wiley/IEEE Press books on mobile radio communications and published in excess of 900 research papers. He was awarded a number of distinctions, most recently the 2007 WCNC best paper award, a best-paper award at ICC'2009, the 2007 IEEE/ComSoc Wireless Technical Commitee Achievement Award and the IET's Sir Monti Finniston Award for contibutions across the entire field of engineering. His current teaching and research interests cover the range of **Mobile Multimedia Communications**, including voice, audio, video and graphical source compression, channel coding, modulation, networking as well as the joint optimisation of these system components. He is managing a research group in the wide field of wireless multimedia communications funded by the Engineering and Physical Sciences Reseach Council (EPSRC), the Commission of European Communities (CEC) and the Virtual Centre of Excellence in Mobile Communications known as MVCE. Lajos is a Fellow of the IEEE and the IEE as well as an IEEE Distinguished Lecturer of both the ComSoc and VTS. He was appointed the Editor-in-Chief of the IEEE Press and paved the way for the joint IEEE Press/Wiley imprint books to appear in IEEE Xplore V.3.0 during 2009.

T.H. Liew received the B.Eng degree in Electronics from the University of Southampton, UK and in 2001 he was awarded the PhD degree. Followsing a one-year spell as a postdoctoral research fellow, he joined Ubinetics in Cambridge, UK. He then transferred to TTP Comms. Ltd in Cambridge and currently working for Motorola. His research interests are associated with coding and modulation for wireless channels, space-time coding and adaptive transceivers. He published his research results widely.

Bee Leong Yeap graduated in Electronics Engineering from the University of Southampton, UK, with a first class honours degree in 1996. In 2000, he was awarded the PhD degree and then continued his research as a postdoctoral research fellow in Southampton for a period of three years. He then joined Radioscape in London, UK, where he designed the signal processing algorithms of DAB transceivers. Recently he joined Motorola in the UK. His research interests include turbo coding, turbo equalisation, adaptive modulation and space-time coding.

Other Related Wiley and IEEE Press Books

- R. Steele, L. Hanzo (Ed): *Mobile Radio Communications: Second and Third Generation Cellular and WATM Systems*, John Wiley and IEEE Press, 2nd edition, 1999, ISBN 07 273-1406-8, 1064 pages
- L. Hanzo, F.C.A. Somerville, J.P. Woodard: *Voice Compression and Communications: Principles and Applications for Fixed and Wireless Channels*; IEEE Press and John Wiley, 2001, 642 pages
- L. Hanzo, P. Cherriman, J. Streit: *Wireless Video Communications: Second to Third Generation and Beyond*, IEEE Press and John Wiley, 2001, 1093 pages
- L. Hanzo, T.H. Liew, B.L. Yeap: *Turbo Coding, Turbo Equalisation and Space-Time Coding*, John Wiley and IEEE Press, 2002, 751 pages
- J.S. Blogh, L. Hanzo: *Third-Generation Systems and Intelligent Wireless Networking: Smart Antennas and Adaptive Modulation*, John Wiley and IEEE Press, 2002, 408 pages
- L. Hanzo, C.H. Wong, M.S. Yee: *Adaptive Wireless Transceivers: Turbo-Coded, Turbo-Equalised and Space-Time Coded TDMA, CDMA and OFDM Systems*, John Wiley and IEEE Press, 2002, 737 pages
- L. Hanzo, L-L. Yang, E-L. Kuan, K. Yen: *Single- and Multi-Carrier CDMA: Multi-User Detection, Space-Time Spreading, Synchronisation, Networking and Standards*, John Wiley and IEEE Press, June 2003, 1060 pages
- L. Hanzo, M. Münster, T. Keller, B-J. Choi, *OFDM and MC-CDMA for Broadband Multi-User Communications, WLANs and Broadcasting*, John-Wiley and IEEE Press, 2003, 978 pages
- L. Hanzo, S-X. Ng, T. Keller and W.T. Webb, *Quadrature Amplitude Modulation: From Basics to Adaptive Trellis-Coded, Turbo-Equalised and Space-Time Coded OFDM, CDMA and MC-CDMA Systems*, John Wiley and IEEE Press, 2004, 1105 pages
- L. Hanzo, T. Keller: *An OFDM and MC-CDMA Primer*, John Wiley and IEEE Press, 2006, 430 pages
- L. Hanzo, F.C.A. Somerville, J.P. Woodard: *Voice and Audio Compression for Wireless Communications*, John Wiley and IEEE Press, 2007, 858 pages
- L. Hanzo, P.J. Cherriman, J. Streit: *Video Compression and Communications: H.261, H.263, H.264, MPEG4 and HSDPA-Style Adaptive Turbo-Transceivers* John Wiley and IEEE Press, 2007, 680 pages
- L. Hanzo, J.S. Blogh, S. Ni: 3G, HSPA and FDD Versus TDD Networking: Smart Antennas and Adaptive Modulation, John Wiley and IEEE Press, February 2008, 596 pages

- L. Hanzo, O. Alamri, M. El-Hajjar, N. Wu: Near-Capacity Multi-Functional MIMOs: Sphere-Packing and Cooperative Space-Time Processing, IEEE Press - John Wiley, February 2009

Acknowledgments

We are indebted to our many colleagues who have enhanced our understanding of the subject, in particular to Prof. Emeritus Raymond Steele. These colleagues and valued friends, too numerous to be mentioned, have influenced our views concerning various aspects of wireless multimedia communications. We thank them for the enlightenment gained from our collaborations on various projects, papers and books. We are grateful to Steve Braithwaite, Jan Brecht, Jon Blogh, Marco Breiling, Marco del Buono, Sheng Chen, Peter Cherriman, Stanley Chia, Byoung Jo Choi, Joseph Cheung, Sheyam Lal Dhomeja, Dirk Didascalou, Lim Dongmin, Stephan Ernst, Peter Fortune, Eddie Green, David Greenwood, Hee Thong How, Thomas Keller, Ee Lin Kuan, W. H. Lam, C. C. Lee, Xiao Lin, Chee Siong Lee, Tong-Hooi Liew, Matthias Münster, Vincent Roger-Marchart, Jason Ng, Michael Ng, M. A. Nofal, Jeff Reeve, Redwan Salami, Clare Somerville, Rob Stedman, David Stewart, Jürgen Streit, Jeff Torrance, Spyros Vlahoyiannatos, William Webb, Stephan Weiss, John Williams, Jason Woodard, Choong Hin Wong, Henry Wong, James Wong, Lie-Liang Yang, Bee-Leong Yeap, Mong-Suan Yee, Kai Yen, Andy Yuen, and many others with whom we enjoyed an association.

We also acknowledge our valuable associations with the Virtual Centre of Excellence (VCE) in Mobile Communications, in particular with its chief executive, Dr Walter Tuttlebee, and other leading members of the VCE, namely Dr Keith Baughan, Prof. Hamid Aghvami, Prof. Ed Candy, Prof. John Dunlop, Prof. Barry Evans, Prof. Peter Grant, Dr Mike Barnard, Prof. Joseph McGeehan, Dr Steve McLaughlin and many other valued colleagues. Our sincere thanks are also due to the EPSRC, UK for supporting our research. We would also like to thank Dr Joao Da Silva, Dr Jorge Pereira, Dr Bartholome Arroyo, Dr Bernard Barani, Dr Demosthenes Ikonomou, Dr Fabrizio Sestini and other valued colleagues from the Commission of the European Communities, Brussels, Belgium, as well as Andy Aftelak, Mike Philips, Andy Wilton, Luis Lopes and Paul Crichton from Motorola ECID, Swindon, UK, for sponsoring some of our recent research. Further thanks are due to Tim Wilkinson and Ian Johnson at HP in Bristol, UK for funding some of our research efforts.

We feel particularly indebted to Rita Hanzo as well as Denise Harvey for their skilful assistance in typesetting the manuscript in LaTeX. Without the kind support of Mark Hammond, Sarah Hinton, Zöe Pinnock and their colleagues at the Wiley editorial office in Chichester, UK this monograph would never have materialised. Finally, our sincere gratitude is due to the numerous authors listed in the Author Index — as well as to those whose work was not cited due to space limitations — for their contributions to the state of the art, without whom this book would not have materialised.

L. Hanzo, TH Liew, BL Yeap and RYS Tee
School of Electronics and Computer Science
University of Southampton

Chapter 1

Historical Perspective, Motivation and Outline[1]

1.1 A Historical Perspective on Channel Coding

The history of channel coding or Forward Error Correction (FEC) coding dates back to Shannon's pioneering work [1] in 1948, predicting that arbitrarily reliable communications are achievable with the aid of channel coding, upon adding redundant information to the transmitted messages. However, Shannon refrained from proposing explicit channel coding schemes for practical implementations. Furthermore, although the amount of redundancy added increases as the associated information delay increases, he did not specify the maximum delay that may have to be tolerated, in order to be able to communicate near the Shannonian limit. In recent years researchers have been endeavouring to reduce the amount of latency inflicted for example by a turbo codec's interleaver that has to be tolerated for the sake of attaining a given target performance.

Historically, one of the first practical FEC codes was the single error correcting Hamming code [2], which was a block code proposed in 1950. Convolutional FEC codes date back to 1955 [3], which were discovered by Elias, while Wozencraft and Reiffen [4,5], as well as Fano [6] and Massey [7], proposed various algorithms for their decoding. A major milestone in the history of convolutional error correction coding was the invention of a maximum likelihood sequence estimation algorithm by Viterbi [8] in 1967. A classic interpretation of the Viterbi Algorithm (VA) can be found, for example, in Forney's often-quoted paper [9]. One of the first practical applications of convolutional codes was proposed by Heller and Jacobs [10] during the 1970s.

We note here that the VA does not result in minimum Bit Error Rate (BER), rather it finds the most likely sequence of transmitted bits. However, it performs close to the minimum possible BER, which can be achieved only with the aid of an extremely complex full-search algorithm evaluating the probability of all possible 2^n binary strings of a k-bit message. The minimum BER decoding algorithm was proposed in 1974 by Bahl *et al.* [11], which was termed the Maximum A-Posteriori (MAP) algorithm. Although the MAP algorithm slightly outperforms the VA in BER terms, because of its significantly higher complexity it was rarely used in practice, until turbo codes were contrived by Berrou *et al.* in 1993 [12,13].

Focusing our attention on block codes, the single error correcting Hamming block code was too weak for practical applications. An important practical milestone was the discovery of the family of multiple error correcting Bose–Chaudhuri–Hocquenghem (BCH) binary block codes [14] in 1959 and in 1960 [15,16]. In

[1] *Turbo Coding, Turbo Equalisation and Space-Time Coding*
L.Hanzo, T.H. Liew, B.L. Yeap,
 ISBN 0-470-84726-3

1960, Peterson [17] recognised that these codes exhibit a cyclic structure, implying that all cyclically shifted versions of a legitimate codeword are also legitimate codewords. The first method for constructing trellises for linear block codes was proposed by Wolf [18] in 1978. Owing to the associated high complexity, there was only limited research in trellis decoding of linear block codes [19, 20]. It was in 1988, when Forney [21] showed that some block codes have relatively simple trellis structures. Motivated by Forney's work, Honary, Markarian and Farrell *et al.* [19,22–25] as well as Lin and Kasami *et al.* [20,26,27] proposed various methods for reducing the associated complexity. The Chase algorithm [28] is one of the most popular techniques proposed for near maximum likelihood decoding of block codes.

Furthermore, in 1961 Gorenstein and Zierler [29] extended the binary coding theory to treat non-binary codes as well, where code symbols were constituted by a number of bits, and this led to the birth of burst-error correcting codes. They also contrived a combination of algorithms, which is referred to as the Peterson–Gorenstein–Zierler (PGZ) algorithm. In 1960 a prominent non-binary subset of BCH codes was discovered by Reed and Solomon [30]; they were named Reed–Solomon (RS) codes after their inventors. These codes exhibit certain optimality properties, since their codewords have the highest possible minimum distance between the legitimate codewords for a given code rate. This, however, does not necessarily guarantee attaining the lowest possible BER. The PGZ decoder can also be invoked for decoding non-binary RS codes. A range of powerful decoding algorithms for RS codes was found by Berlekamp [31, 32] and Massey [33, 34]. Various soft-decision decoding algorithms were proposed for the soft decoding of RS codes by Sweeney [35–37] and Honary [19]. In recent years RS codes have found practical applications, for example, in Compact Disc (CD) players, in deep-space scenarios [38], and in the family of Digital Video Broadcasting (DVB) schemes [39], which were standardised by the European Telecommunications Standardisation Institute (ETSI).

Inspired by the ancient theory of Residue Number Systems (RNS) [40–42], which constitute a promising number system for supporting fast arithmetic operations [40, 41], a novel class of non-binary codes referred to as Redundant Residue Number System (RRNS) codes were introduced in 1967. An RRNS code is a maximum–minimum distance block code, exhibiting similar distance properties to RS codes. Watson and Hastings [42] as well as Krishna *et al.* [43, 44] exploited the properties of the RRNS for detecting or correcting a single error and also for detecting multiple errors. Recently, the soft decoding of RRNS codes was proposed in [45].

During the early 1970s, FEC codes were incorporated in various deep-space and satellite communications systems, and in the 1980s they also became common in virtually all cellular mobile radio systems. However, for a long time FEC codes and modulation have been treated as distinct subjects in communication systems. By integrating FEC and modulation, in 1987 Ungerboeck [46–48] proposed Trellis Coded Modulation (TCM), which is capable of achieving significant coding gains over power and band-limited transmission media. A further historic breakthrough was the invention of turbo codes by Berrou, Glavieux, and Thitimajshima [12, 13] in 1993, which facilitate the operation of communications systems near the Shannonian limits. Turbo coding is based on a composite codec constituted by two parallel concatenated codecs. Since its recent invention turbo coding has evolved at an unprecedented rate and has reached a state of maturity within just a few years due to the intensive research efforts of the turbo coding community. As a result of this dramatic evolution, turbo coding has also found its way into standardised systems, such as for example the recently ratified third-generation (3G) mobile radio systems [49]. Even more impressive performance gains can be attained with the aid of turbo coding in the context of video broadcast systems, where the associated system delay is less critical than in delay-sensitive interactive systems.

More specifically, in their proposed scheme Berrou *et al.* [12, 13] used a parallel concatenation of two Recursive Systematic Convolutional (RSC) codes, accommodating the turbo interleaver between the two encoders. At the decoder an iterative structure using a modified version of the classic minimum BER MAP invented by Bahl *et al.* [11] was invoked by Berrou *et al.*, in order to decode these parallel concatenated codes. Again, since 1993 a large amount of work has been carried out in the area, aiming for example to reduce the associated decoder complexity. Practical reduced-complexity decoders are for example the Max-Log-MAP algorithm proposed by Koch and Baier [50], as well as by Erfanian *et al.* [51], the Log-MAP algorithm suggested by Robertson, Villebrun and Hoeher [52], and the SOVA advocated by Hagenauer as well as Hoeher [53,54]. Le Goff, Glavieux and Berrou [55], Wachsmann and Huber [56] as well as Robertson and Worz [57] suggested the use of these codes in conjunction with bandwidth-efficient modulation schemes. Further advances in understanding the excellent performance of the codes are due, for example, to Benedetto and Montorsi [58,59] and Perez, Seghers and Costello [60]. During the mid-1990s Hagenauer, Offer and Papke [61], as well as Pyndiah [62], extended the turbo concept to parallel concatenated block codes as well. Nickl *et al.* show in [63] that Shannon's limit can be approached within 0.27 dB by employing a simple turbo Hamming code. In [64] Acikel and Ryan proposed an efficient procedure for designing the puncturing patterns for high-rate turbo convolutional codes. Jung and Nasshan [65, 66] characterised the achievable turbo-coded performance under the constraints of short transmission frame lengths, which is characteristic of interactive speech systems. In collaboration with Blanz they also applied turbo codes to a CDMA system using joint detection and antenna diversity [67]. Barbulescu and Pietrobon addressed the issues of interleaver design [68]. The tutorial paper by Sklar [69] is also highly recommended as background reading.

Driven by the urge to support high data rates for a wide range of bearer services, Tarokh, Seshadri and Calderbank [70] proposed space-time trellis codes in 1998. By jointly designing the FEC, modulation, transmit diversity and optional receive diversity scheme, they increased the throughput of band-limited wireless channels. A few months later, Alamouti [71] invented a low-complexity space-time block code, which offers significantly lower complexity at the cost of a slight performance degradation. Alamouti's invention motivated Tarokh *et al.* [72, 73] to generalise Alamouti's scheme to an arbitrary number of transmitter antennas. Then, Tarokh *et al.*, Bauch *et al.* [74, 75], Agrawal *et al.* [76], Li *et al.* [77, 78] and Naguib *et al.* [79] extended the research of space-time codes from considering narrowband channels to dispersive channels [70, 71, 73, 79, 80].

1.1.1 A Historical Perspective on Coded Modulation

When using separate coding and modulation, achieving a close-to-Shannon-limit performance often requires a low coding rate, hence resulting in a high bandwidth expansion. Therefore, a bandwidth efficient Multilevel Coding (MLC) scheme, which was based on the joint design of coding and modulation, was proposed by Imai and Hirawaki [81] in 1977. This scheme employed several component codes and invoked a MultiStage Decoding (MSD) method, where the redundant FEC bits may be absorbed without bandwidth expansion by expanding the modulated phasor constellation. This multistage decoding procedure was further investigated by Calderbank in [82].

Ungerböck's concept of Trellis Coded Modulation (TCM) was independently proposed in 1982, which amalgamated the design of coding and modulation into a single entity with the aid of Ungerböck's constellation partitioning [83]. MLC based on Ungerböck's partitioning of the modulated signal sets were also studied by Pottie $et\ al.$ [86]. The performance of MLCs and TCM in Gaussian channels was further investigated by Kofman $et\ al.$ in [88], when using interleavers and limited soft-output based MSD. The MLC aided TCM design constructed with the aid of convolutional codes having maximum Hamming distance for transmission over Rayleigh fading channel was presented in [87]. The specific rate of the individual component codes of MLCs designed for approaching the capacity was determined by Huber $et\ al.$ [89]. The provision of Unequal Error Protection (UEP) is important in multimedia transmissions, hence Lin $et\ al.$ [93, 94] designed UEP aided MLCs for both symmetric and asymmetric constellations.

In order to exploit the powerful error correction capability of LDPCs, Hou $et\ al.$ employed them as MLC component codes and designed power- and bandwidth-efficient MLC schemes for Code Devision Multiple Access (CDMA) [97]. In order to obtain a further diversity gain with the aid of multiple transmit and receive antennas, Lampe $et\ al.$ proposed a multiple-antenna assisted transmission scheme for MLCs [98]. The employment of Multi-Dimensional (M-D) space-time MLCs involving M-D partitioning was carried out in the work of Martin $et\ al.$ [99] in order to obtain substantial coding gains.

In 1982, the TCM concept was adopted by Zehavi to improve the achievable time-diversity order while maintaining a minimum Hamming distance, which led to the concept of Bit-Interleaved Coded Modulation (BICM) [84]. This improved the achievable coded modulation performance, when communicating over Rayleigh fading channels. The theory underlying BICM was extensively analysed by Caire $et\ al.$ in terms of its channel capacity, error bound and design guidelines in [91]. For the sake of introducing the iterative decoding of BICM and hence achieving an improved performance in Additive White Gaussian Channels

Year	Author(s)	Contribution
1977	Imai and Hirawaki [81]	Proposed MLC invoking multistage decoding.
1982	Ungerböck [83]	Invented TCM employing Ungerböck partitioning (UP).
1982	Zehavi [84]	Invented BICM for transmission over Rayleigh fading channels.
1987	Wei [85]	Pioneered rotationally invariant differentially encoded multidimensional constellation for the design of TCM.
1989	Calderbank [82]	Investigated MSD aided MLC.
1989	Pottie and Taylor [86]	Designed MLC based on UP strategies.
1993	Seshadri and Sundberg [87]	Studied the performance of Multilevel TCM in Rayleigh fading channel.
1994	Kofman [88]	Studied the performance of MLC in AWGN channels.
1994	Huber and Wachsmann [89]	Calculated the equivalent capacity of MLC schemes.
1997	Li and Ritcey [90]	Designed BICM-ID using UP strategy.
1998	Caire $et\ al.$ [91]	Analysed the theoretical error bound of BICM.
1998	Robertson $et\ al.$ [92]	Designed iterative turbo-detection aided TTCM.
2000	Shu Lin $et\ al.$ [93, 94]	Designed UEP for MLC based on symmetrical and asymmetrical phasor constellations.
2001	Ritcey $et\ al.$ [95]	Introduced improved bit-to-symbol mapping for BICM-ID.
2004	Huang $et\ al.$ [96]	Designed improved mapping schemes for space-time BICM-ID.
2004	Hou $et\ al.$ [97]	Employed LDPC as MLC component codes and introduced a novel semi-BICM structure.
2004	Lampe $et\ al.$ [98]	Proposed MLC-aided multiple-antenna assisted transmission schemes.
2006	Martin $et\ al.$ [99]	Devised an MLC based multidimensional mapping scheme for space-time codes.
2007	Mohammed $et\ al.$ [100, 101]	Introduced multidimensional mapping for space-time BICM-ID employing the Reactive Tabu Search technique.
2007	Matsumoto $et\ al.$ [102]	Introduced an adaptive coding technique for multilevel BICM aided broad-band single carrier signaling.
2008	Simoens $et\ al.$ [103]	Investigated the effects of linear precoding on BICM-ID for transmission over AWGN channels.

Table 1.1: History of coded modulation contributions.

(AWGN), the Bit-Interleaved Coded Modulation based Iterative Decoding (BICM-ID) philosophy was proposed by Li $et\ al.$ [90] using the Ungerböck's TCM partitioning strategy.

The multidimensional TCM concept was pioneered by Wei [85] for the sake of achieving rotational invariance which has the potential of dispensing with the false locking problems of carrier recovery as well as the concomitant avalanche-like error propagation. To introduce iterative decoding, two parallel TCM schemes were invoked by Robertson $et\ al.$ in [92]. This parallel concatenated design was later termed as Turbo TCM (TTCM).

Since the optimisation of the bit-to-symbol mapping for BICM-ID was found to be crucial in assisting the scheme's iterative decoding convergence, Ritcey $et\ al.$ further improved the mapping [95] schemes. BICM-ID was combined with space time codes to achieve a spatial diversity gain and the corresponding mapping schemes were further improved by Huang $et\ al.$. Mohammed $et\ al.$ [100] later extended the findings of [96] to multidimensional constellation labelling by employing the Reactive Tabu Search (RTS) technique [104].

A MultiLevel BICM scheme (ML-BICM) was combined with ARQ and adaptive coding in the work of Matsumoto [102] $et\ al.$. This flexible design could be viewed as layer-by-layer link adaptation combined with an effective retransmission scheme. The multidimensional mapping used could be interpreted as a combined mapping function, a serially concatenated rate-one precoder and a Gray mapper. Simoens $et\ al.$ [103] investigated the optimised linear block precoder aided design of BICM-ID communicating over AWGN channels for the sake of achieving an infinitesimally low BER.

The major contributions of the past three decades in the field of coded modulation, particularly in multilevel and bit-interleaved coded modulation are summarised in Table 1.1.

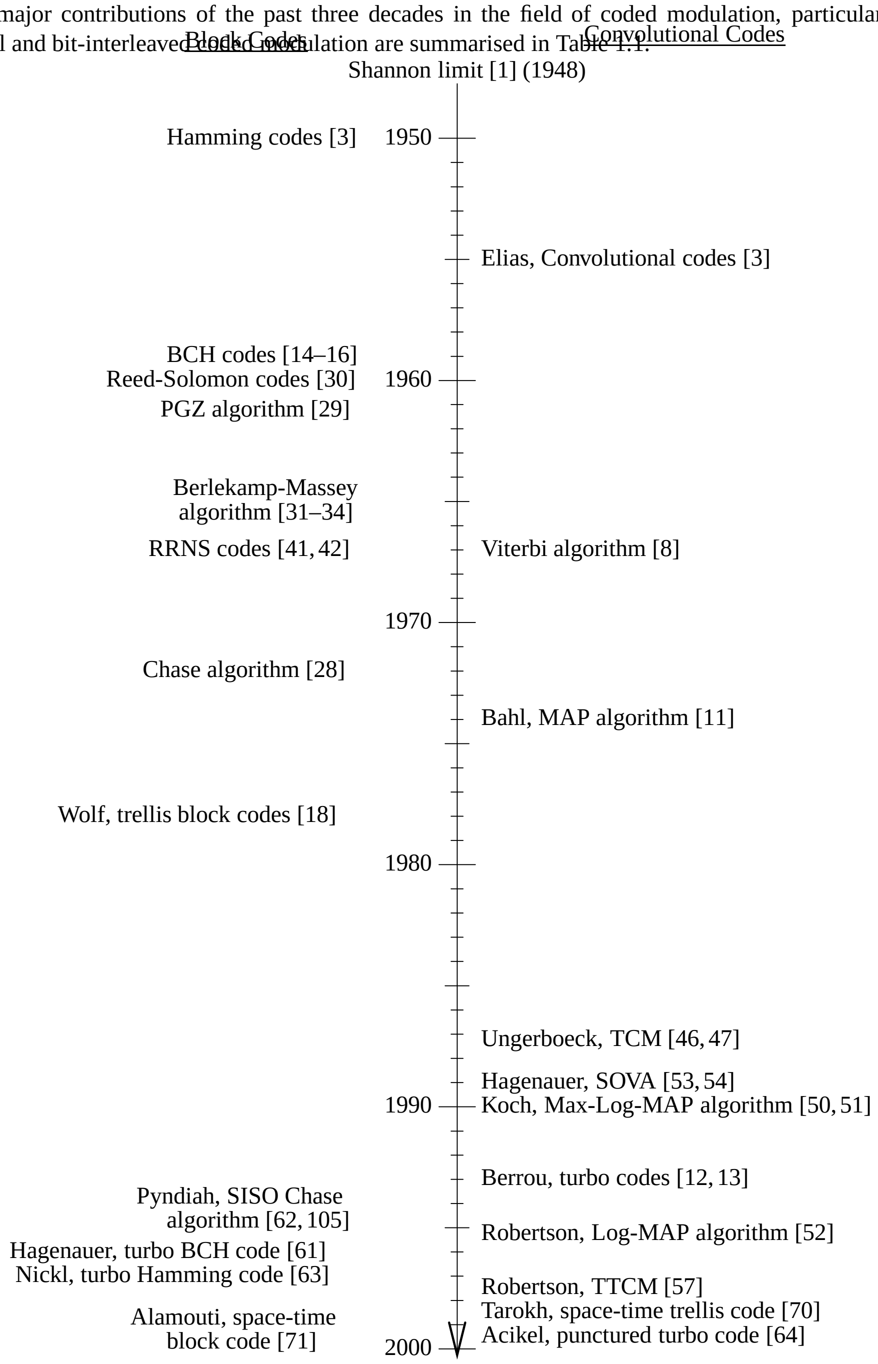


Figure 1.1: A brief history of channel coding.

In Figure 1.1, we show the evolution of channel coding research over the past 50 years since Shannon's legendary contribution [1]. These milestones have been incorporated also in the range of monographs and textbooks summarised in Figure 1.2. At the time of writing, the Shannon limit may be approached within a tiny margin [63], provided that the associated decoding complexity and delay is deemed affordable. The challenge is to contrive FEC schemes which are capable of achieving a performance near the *capacity of wireless channels* at an affordable delay and decoding complexity.

1.2 Motivation of the Book

The design of an attractive channel coding and modulation scheme depends on a range of contradictory factors, which are portrayed in Figure 1.3. The message of this illustration is multi-fold. For example, given a certain transmission channel, it is always feasible to design a coding and modulation ('codulation') system, which can further reduce the BER achieved. This typically implies, however, further investments and/or penalties in terms of the required increased implementational complexity and coding/interleaving delay as well as reduced effective throughput. Different solutions accrue when optimising different codec features. For example, in many applications the most important codec parameter is the achievable coding gain, which quantifies the amount of bit-energy reduction attained by a codec at a certain target BER. Naturally, transmitted power reduction is extremely important in battery-powered devices. This transmitted power reduction is only achievable at the cost of an increased implementational complexity, which itself typically increases the power consumption and hence erodes some of the power gain.

Viewing this system optimisation problem from a different perspective, it is feasible to transmit at a higher bit rate in a given fixed bandwidth by increasing the number of bits per modulated symbol. However, when aiming for a given target BER, the channel coding rate has to be reduced, in order to increase the transmission integrity. Naturally, this reduces the *effective throughput* of the system and results in an overall increased system complexity. When the channel's characteristic and the associated bit error statistics change, different solutions may become more attractive. This is because Gaussian channels, narrowband and wideband Rayleigh fading or various Nakagami fading channels inflict different impairments. These design trade-offs constitute the subject of this monograph.

Our intention with the book is multi-fold:

1) *First, we would like to pay tribute to all researchers, colleagues and valued friends who contributed to the field.* Hence this book is dedicated to them, since without their quest for better coding solutions to communications problems this monograph could not have been conceived. They are too numerous to name here, hence they appear in the author index of the book.

2) The invention of turbo coding not only assisted in attaining a performance approaching the Shannonian limits of channel coding for transmissions over **Gaussian channels**, but also revitalised channel coding research. In other words, turbo coding opened a new chapter in the design of iterative detection-assisted communications systems, such as turbo trellis coding schemes, turbo channel equalisers, etc. Similarly dramatic advances have been attained with the advent of space-time coding, when communicating over dispersive, fading **wireless channels**. *Recent trends indicate that better overall system performance may be attained by jointly optimising a number of system components, such as channel coding, channel equalisation, transmit and received diversity and the modulation scheme, than in case of individually optimising the system components. This is the main objective of this monograph.*

3) *Since at the time of writing no joint treatment of the subjects covered by this book exists, it is timely to compile the most recent advances in the field. Hence it is our hope that the conception of this monograph on the topic will present an adequate portrayal of the last decade of research and spur this innovation process by stimulating further research in the coding and communications community.*

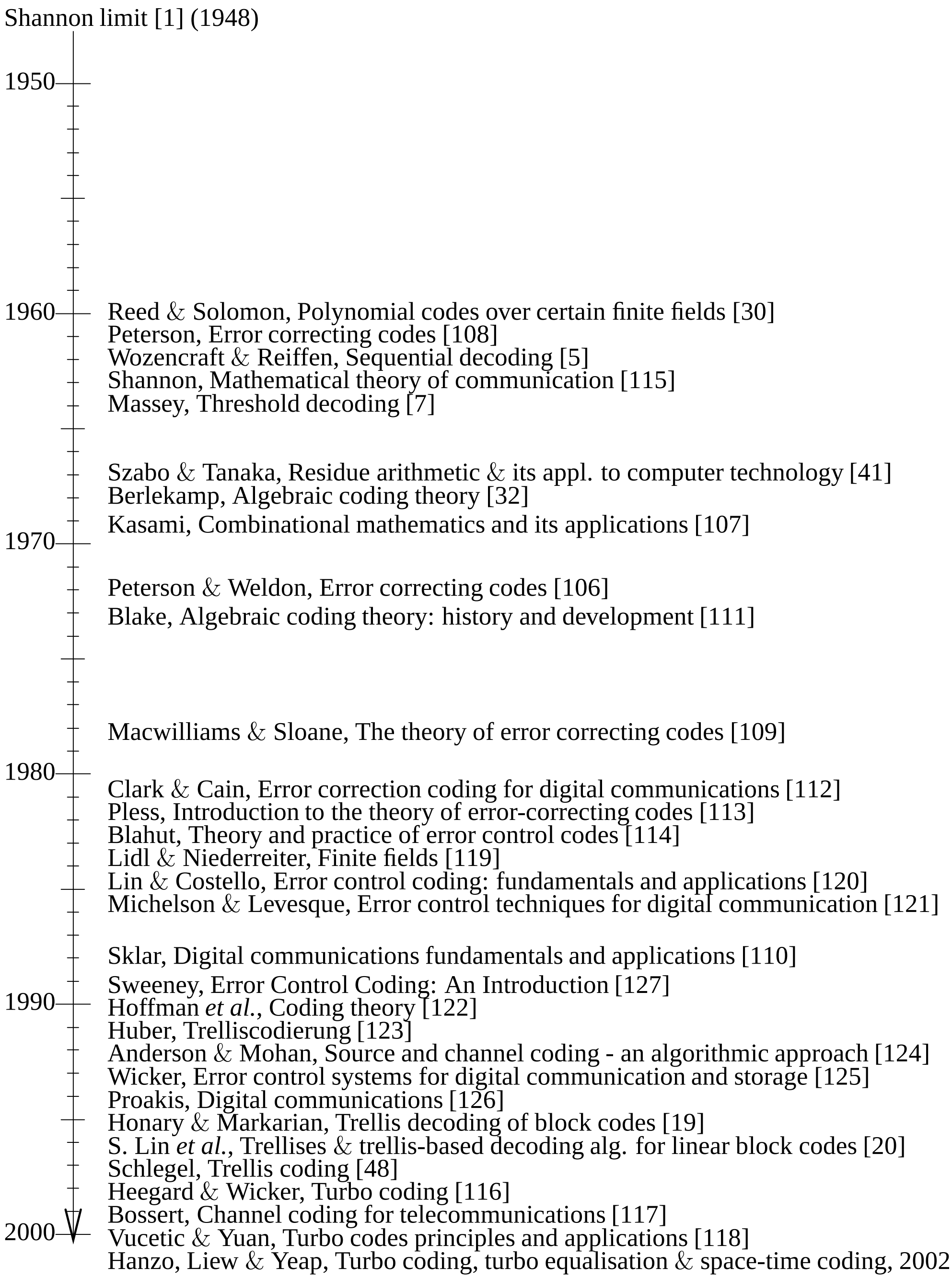


Figure 1.2: Milestones in channel coding.

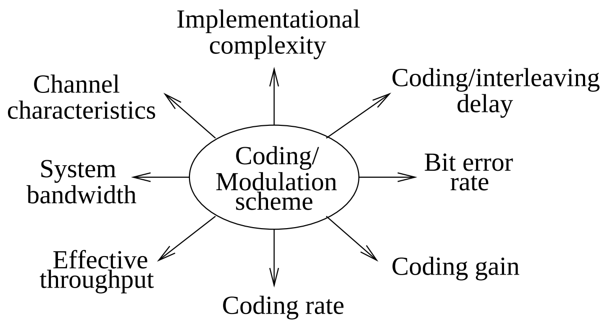


Figure 1.3: Factors affecting the design of channel coding and modulation scheme.

1.3 Organisation of the Book

Below, we present the outline and rationale of the book:

- **Chapter 2:** For the sake of completeness and wider reader appeal virtually no prior knowledge is assumed in the field of channel coding. Hence in Chapter 2 we commence our discourse by introducing the family of convolutional codes and the hard- as well as soft-decision Viterbi algorithm in simple conceptual terms with the aid of worked examples.
- **Chapter ??:** This chapter provides a rudimentary introduction to the most prominent classes of block codes, namely to Reed–Solomon (RS) and Bose–Chaudhuri–Hocquenghem (BCH) codes. A range of algebraic decoding techiques are also reviewed and worked examples are included.
- **Chapter ??:** Based on the simple Viterbi decoding concepts introduced in Chapter 2, in this chapter an overview of the family of conventional binary BCH codes is given, with special emphasis on their trellis decoding. In parallel to our elaborations in Chapter 2 on the context of convolutional codes, the Viterbi decoding of binary BCH codes is detailed with the aid of worked examples. These discussions are followed by the simulation-based performance characterisation of various BCH codes employing both hard-decision and soft-decision decoding methods. The classic Chase algorithm is introduced and its performance is investigated.
- **Chapter 3:** This chapter introduces the concept of turbo convolutional codes and gives a detailed discourse on the Maximum A-Posteriori (MAP) algorithm and its computationally less demanding counterparts, namely the Log-MAP and Max-Log-MAP algorithms. The Soft-Output Viterbi Algorithm (SOVA) is also highlighted and its concept is augmented with the aid of a detailed worked example. Then the effects of the various turbo codec parameters are investigated, namely that of the number of iterations, the puncturing patterns used, the component decoders, the influence of the interlever depth, which is related to the codeword length, etc. The various codecs' performance is studied also when communicating over Rayleigh fading channels.
- **Chapter 4:** The concept of turbo codes using BCH codes as component codes is introduced. A detailed derivation of the MAP algorithm is given, building on the concepts introduced in Chapter 3 in the context of convolutional turbo codes, but this time cast in the framework of turbo BCH codes. Then, the MAP algorithm is modified in order to highlight the concept of the Max-Log-MAP and Log-MAP algorithms, again, with reference to binary turbo BCH codes. Furthermore, the SOVA-based binary BCH decoding algorithm is introduced. Then a simple turbo decoding example is given, highlighting how iterative decoding assists in correcting multiple errors. We also describe a novel MAP algorithm for decoding extended BCH codes. Finally, we show the effects of the various coding parameters on the performance of turbo BCH codes.
- **Chapter 5:** Space-time block codes are introduced. The derivation of the MAP decoding of space-time block codes is then given. A system is proposed by concatenating space-time block codes

and various channel codes. The complexity and memory requirements of various channel decoders are derived, enabling us to compare the performance of the proposed channel codes by considering their decoder complexity. Our simulation results related to space-time block codes using no channel coding are presented first. Then, we investigate the effect of mapping data and parity bits from binary channel codes to non-binary modulation schemes. Finally, we compare our simulation results for various channel codes concatenated with a simple space-time block code. Our performance comparisons are conducted by also considering the complexity of the associated channel decoder.

- **Chapter 6:** The encoding process of space-time trellis codes is highlighted. This is followed by employing an Orthogonal Frequency Division Multiplexing (OFDM) modem in conjunction with space-time codes over wideband channels. Turbo codes and RS codes are concatenated with space-time codes in order to improve their performance. Then, the performance of the advocated space-time block code and space-time trellis codes is compared. Their complexity is also considered in comparing both schemes. The effect of delay spread and maximum Doppler frequency on the performance of the space-time codes is investigated. A Signal to Interference Ratio (SIR) related term is defined in the context of dispersive channels for the advocated space-time block code, and we will show how the SIR affects the performance of the system. In our last section, we propose space-time-coded Adaptive OFDM (AOFDM). We then show by employing multiple antennas that with the advent of space-time coding, the wideband fading channels have been converted to AWGN-like channels.

- **Chapter 7:** The discussions of Chapters 5 and 6 were centred around the topic of employing multiple-transmitter, multiple-receiver (MIMO) based transmit and receive-diversity assisted space-time coding schemes. These arrangements have the potential of significantly mitigating the hostile fading wireless channel's near-instantaneous channel quality fluctuations. Hence these space-time codecs can be advantageously combined with powerful channel codecs originally designed for Gaussian channels. As a lower-complexity design alternative, this chapter introduces the concept of near-instantaneously Adaptive Quadrature Amplitude Modulation (AQAM), combined with near-instantaneously adaptive turbo channel coding. These adaptive schemes are capable of mitigating the wireless channel's quality fluctuations by near-instantaneously adapting both the modulation mode used as well as the coding rate of the channel codec invoked. The design and performance study of these novel schemes constitutes the topic of Chapter 7.

- **Chapter 8:** This chapter focuses on the portrayal of partial-response modulation schemes, which exhibit impressive performance gains in the context of joint iterative, joint channel equalisation and channel decoding. This joint iterative receiver principle is termed turbo equalisation. An overview of Soft-In/Soft-Out (SISO) algorithms, namely that of the MAP algorithm and Log-MAP algorithm, is presented in the context of GMSK channel equalisation, since these algorithms are used in the investigated joint channel equaliser and turbo decoder scheme.

- **Chapter 9:** Based on the introductory concepts of Chapter 8, in this chapter the detailed principles of iterative joint channel equalisation and channel decoding techniques known as turbo equalisation are introduced. This technique is invoked in order to overcome the unintentional Inter-Symbol Interference (ISI) and Controlled Inter-Symbol Interference (CISI) introduced by the channel and the modulator, respectively. Modifications of the SISO algorithms employed in the equaliser and decoder are also portrayed, in order to generate information related not only to the source bits but also to the parity bits of the codewords. The performance of coded systems employing turbo equalisation is analysed. Three classes of encoders are utilised, namely convolutional codes, convolutional-coding-based turbo codes and BCH-coding-based turbo codes.

- **Chapter ??:** Theoretical models are devised for the coded schemes in order to derive the maximum likelihood bound of the system. These models are based on the Serial Concatenated Convolutional Code (SCCC) analysis presented in reference [128]. Essentially, this analysis can be employed since the modulator could be represented accurately as a rate $R = 1$ convolutional encoder. Apart from convolutional-coded systems, turbo-coded schemes are also considered. Therefore the theoretical concept of Parallel Concatenated Convolutional Codes (PCCC) [59] is utilised in conjunction with the SCCC principles in order to determine the Maximum Likelihood (ML) bound of the turbo-coded systems, which are modelled as hybrid codes consisting of a parallel concatenated convolutional code, serially linked with another convolutional code. An abstract interleaver from reference [59] — termed the uniform interleaver — is also utilised, in order to reduce the complexity associated with determining all the possible interleaver permutations.

- **Chapter 10:** A comparative study of coded BPSK systems, employing high-rate channel encoders, is presented. The objective of this study is to investigate the performance of turbo equalisers in systems employing different classes of codes for high code rates of $R = \frac{3}{4}$ and $R = \frac{5}{6}$, since known turbo equalisation results have only been presented for turbo equalisers using convolutional codes and convolutional-based turbo codes for code rates of $R = \frac{1}{3}$ and $R = \frac{1}{2}$ [129, 130]. Specifically, convolutional codes, convolutional-coding-based turbo codes, and Bose–Chaudhuri–Hocquengham (BCH)-coding-based [14, 15] turbo codes are employed in this study.

- **Chapter 11:** A novel reduced-complexity trellis-based equaliser is presented. In each turbo equalisation iteration the decoder generates information which reflects the reliability of the source and parity bits of the codeword. With successive iteration, the reliability of this information improves. This decoder information is exploited in order to decompose the received signal such that each quadrature component consists of the in-phase or quadrature-phase component signals. Therefore, the equaliser only has to consider the possible in-phase or quadrature-phase components, which is a smaller set of signals than all of their possible combinations.

- **Chapter 12:** For transmissions over wideband fading channels and fast fading channels, space-time trellis coding (STTC) is a more appropriate diversity technique than space-time block coding. STTC [70] relies on the joint design of channel coding, modulation, transmit diversity and the optional receiver diversity schemes. The decoding operation is performed by using a maximum likelihood detector. This is an effective scheme, since it combines the benefits of Forward Error Correction (FEC) coding and transmit diversity, in order to obtain performance gains. However, the cost of this is the additional computational complexity, which increases as a function of bandwidth efficiency (bits/s/Hz) and the required diversity order. In this chapter STTC is investigated for transmission over wideband fading channels.

- **Chapter 13:** Our previous discussions on various channel coding schemes evolves to the family of joint coding and modulation-based arrangements, which are often referred to as coded modulation schemes. Specifically, Trellis-Coded Modulation (TCM), Turbo Trellis-Coded Modulation (TTCM), Bit-Interleaved Coded Modulation (BICM) as well as iterative joint decoding and demodulation-assisted BICM (BICM-ID) will be studied and compared under various narrowband and wideband propagation conditions.

- **Chapter 14: Multilevel Coding Theory**

  This chapter introduces the background of MLCs. Section 14.2 highlights the design of signal labelling, rate design criteria as well as the encoding and decoding structures. BICM and BICM-ID are also characterised in terms of the philosophy of using bit interleavers, their decoding methods as well as bit-to-symbol mapping schemes in Section 13.5 and 14.4, respectively.

- **Chapter 15: MLC Design Using EXIT Chart Analysis**

  The iterative detection of BICM-ID and MLC schemes is analysed with respect to their convergence behaviour using EXIT chart analysis in this chapter. Section 15.2 presents the comparative study of different coded modulation schemes, namely that of MLC, BICM, BICM-ID, TCM and TTCM as a function of their trellis complexity expressed in terms of the number of trellis states and interleaver length. The simulation results showing these comparisons are presented in Section 15.2.3. EXIT charts employed as a design tool for iterative decoding are described in Section 15.3, characterising the iterative detection aided performance of BICM-ID schemes. Three-dimensional EXIT charts are used for studying the convergence behaviour of MLC schemes in Section 15.3.3. A precoder-aided MLC design is introduced in Section 15.4, which employs the above-mentioned 3-D EXIT analysis. The performance improvements achieved by this system are presented in Section 15.4.3.

- **Chapter 16: Sphere Packing Aided MLC/BICM Space-Time Design**

  This chapter studies the new arrangement of MLC combined with Space Time Block Codes (STBC) invoking a new type of M-D Sphere Packing (SP) modulation scheme termed as a STBC-SP-MLC arrangement. A similar arrangement using BICM to replace the MLC coding block is also used, which is referred to as an STBC-SP-BICM arrangement. Section 16.2 describes Alamouti's twin-antenna based STBC scheme. The SP modulation and its soft demodulation technique are outlined in Sections 16.3 and 16.4. The STBC-SP-BICM arrangement is presented in Section 16.5, where we further highlight the features of the system, as well as its equivalent capacity based design employed for determining the rates of our MLC component codes and the appropriate bit-to-SP-symbol mapping schemes. In Section 16.5.4, the UEP scheme created by this system is investigated by introducing a $hybrid$ bit-stream partitioning strategy. Section 16.6 considers this STBC-SP-BICM arrangement, presenting the general system structure as well as a range of various mapping schemes. EXIT charts are invoked for further analysis both with and without precoder enhancements, and the corresponding simulation results are discussed in Section 16.6.5.

- **Chapter 17: MLC/BICM Schemes for the Wireless Internet**

  This chapter provides MLC designs for low-latency multimedia applications employing Parallel Independent Decoding (PID) aided Generalised Low-Density Parity Check (GLDPC) component codes. Furthermore, BICM schemes invoking Luby-Transform (LT) coding constructed for hostile AWGN-contaminated BEC propagation conditions are investigated in a wireless Internet scenario. The Multilevel GLDPC (MLC-GLDPC) schemes are detailed in Section 17.2. We outline the associated GLDPC structure, together with the MLC-GLDPC encoding and decoding methods, when using BCH codes as constituent codes. Section 17.3 describes the design objectives of invoking both inner and outer iterations. The GLDPC code's syndrome checking method used and the associated simulation results are provided in Sections 17.3.1 and 17.3.2. The design philosophy of Fountain codes and LT codes contrived for the wireless Internet are detailed in Sections 17.4.1 and 17.4.2. Sections 17.4.2.1 and 17.4.2.2 describe the LT code's degree distribution leading to the concept of the $improved\ Robust$ distribution. A novel serially concatenated LT and BICM-ID (LT-BICM-ID) arrangement is presented in Section 17.4.3, outlining the system's construction. In Section 17.5, we enhance the LT-BICM-ID system with the aid of an LLR reliability estimation scheme employed for declaring packet erasure in the amalgamated LT-BICM-ID structure. Section 17.5.4 details the bit-by-bit LT decoding procedures, while our simulation results are discussed in Section 17.5.5.

- **Chapter 18: Near Capacity Irregular BICM-ID Design**

  This chapter outlines the concept of irregular component codes employed in the amalgamated BICM-ID scheme. The resultant scheme is termed as Irregular Bit-Interleaved Coded Modulation using Iterative Decoding (Ir-BICM-ID), which employs three different irregular components, namely Irregular Convolutional Codes (IrCC), Irregular Unity-Rate Codes (IrURC) and Irregular Mappers (IrMapper), for the sake of approaching the theoretical capacity limit. The proposed Ir-BICM-ID scheme is detailed in Section 18.2, where the detailed schematic showing the separate subcodes is shown in Figure 18.3. We characterise the resultant near-capacity scheme using EXIT chart analysis in Section 18.3, demonstrating that it exhibits a narrow but still open tunnel between the outer and inner codes' EXIT functions. Section 18.4.1 introduces an IrCC combined with both different-rate convolutional codes and memoryless repetition codes, in order to create a diverse range of EXIT functions, as illustrated in Figure 18.5. The inner EXIT functions, which are generated using the combination of an IrURC and an IrMapper are illustrated in Section 18.4.2. An appropriate EXIT chart matching algorithm is detailed in Section 18.4.3, while our simulation results characterising the proposed Ir-BICM-ID scheme are discussed in Section 18.5. Finally, we conclude this chapter in Section 18.6.

- **Chapter 19:** This chapter provides a brief summary of the book.

It is our hope that this book portrays the range of contradictory system design trade-offs associated with the conception of channel coding arrangements in an unbiased fashion and that readers will be able to glean information from it in order to solve their own particular channel coding and communications problem.

Most of all, however, we hope that they will find it an enjoyable and informative read, providing them with intellectual stimulation.

1.4 Novel Contributions of the Book

This book is based on a number of related publications of the authors [131–144] as well as on substantial unpublished material. Hence, although it is not customary to list the novel contributions of book, in this research monograph we would like to deviated from this practice and list the novel contributions as follows:

- Different coded modulation schemes, namely the MLC, BICM, BICM-ID, TCM and TTCM, were studied comparatively in terms of their performance versus trellis-complexity, their ability to support unequal error protection classes as well as in terms of the effects of the interleaver length, when fixing the number of iterations. The general structure of a novel 3-D EXIT chart was devised, when using 8-level Phase Shift Keying (8PSK) and three en(de)coders. The 3-D EXIT characteristics are useful for analysing the iterative decoding convergence of three-component MLC schemes using MSD [132].
- Based on the 3-D EXIT charts, a novel precoded-MLC scheme was proposed [135], where the EXIT chart based convergence analysis was facilitated. The precoded-MLC scheme provided substantial BER improvements when transmitting over uncorrelated Rayleigh fading channels.
- In order to introduce transmit diversity, we developed the novel STBC-SP-MLC scheme of Chapter 16 which utilised the M-D SP in Alamouti's twin-antenna based transmitter design. Explicitly, we introduced a novel equivalent capacity based code-rate design for determining the MLC component rates, various bit-to-SP-symbol mapping schemes, a novel Cost Function (CF) for attaining the most appropriate mapping using the Binary Switching Algorithm (BSA) and a $hybrid$ UEP bit-to-SP-symbol mapping design [136, 137, 141].
- A novel STBC-SP-BICM arrangement was proposed [140]. The schemes having $M = 16$ and $M = 256$ SP constellation points were investigated, where the minimum of the CF was found with the aid of the BSA. When designing the bit-to-SP-symbol mapping, various layers of the SP constellation space were used and diverse precoded systems were designed.
- Multilevel schemes invoking GLDPCs as their component codes were proposed [131] using implementationally attractive short BCH and Hamming constituent codes. A novel stopping criterion was also designed for both the inner and outer iterations of the MLC-GLDPC scheme.
- A serially concatenated LT-BICM-ID scheme was proposed for the wireless Internet [134]. To further develop the system, an LLR based packet reliability estimation scheme was presented [138] for the amalgamated LT-BICM-ID design.
- An EXIT-chart aided Ir-BICM-ID design was proposed [142, 143] for the sake of achieving a near-capacity performance. The outer IrCC scheme generates a diverse range of outer EXIT functions, which were closely matched by those of the combination of an IrURC and IrMapper. A novel EXIT function matching algorithm was used for creating a narrow, but still open EXIT tunnel, which led to a near-capacity Ir-BICM-ID scheme.

L. Hanzo, TH. Liew, BL. Yeap, RYS. Tee
School of Electronics and Computer Science
University of Southampton

Part I

Turbo Convolutional and Turbo Block Coding

Chapter 3

Turbo Convolutional Coding [1]

J.P. Woodard, L. Hanzo[2]

3.1 Introduction

In Chapter 2 a rudimentary introduction to convolutional codes and their decoding using the Viterbi algorithm was provided. In this chapter we introduce the concept of turbo coding using convolutional codes as its constituent codes. Our discussions will become deeper, relying on basic familiarity with convolutional coding.

The insightful concept of turbo coding was proposed in 1993 in a seminal contribution by Berrou, Glavieux and Thitimajashima, who reported excellent coding gain results [12], approaching Shannonian predictions. The information sequence is encoded twice, with an interleaver between the two encoders serving to make the two encoded data sequences approximately statistically independent of each other. Often half-rate Recursive Systematic Convolutional (RSC) encoders are used, with each RSC encoder producing a systematic output which is equivalent to the original information sequence, as well as a stream of parity information. The two parity sequences can then be punctured before being transmitted along with the original information sequence to the decoder. This puncturing of the parity information allows a wide range of coding rates to be realised, and often half the parity information from each encoder is sent. Along with the original data sequence this results in an overall coding rate of 1/2.

At the decoder two RSC decoders are used. Special decoding algorithms must be used which accept soft inputs and give soft outputs for the decoded sequence. These soft inputs and outputs provide not only an indication of whether a particular bit was a 0 or a 1, but also a likelihood ratio which gives the probability that the bit has been correctly decoded. The turbo decoder operates iteratively. In the first iteration the first RSC decoder provides a soft output giving an estimation of the original data sequence based on the soft channel inputs alone. It also provides an *extrinsic* output. The extrinsic output for a given bit is based not on the channel input for that bit, but on the information for surrounding bits and the constraints imposed by the code being used. This extrinsic output from the first decoder is used by the second RSC decoder as *a-priori information*, and this information together with the channel inputs are used by the second RSC decoder to give its soft output and extrinsic information. In the second iteration the extrinsic information from the

[1] *Turbo Coding, Turbo Equalisation and Space-Time Coding*
L.Hanzo, T.H. Liew, B.L. Yeap,
©2002 John Wiley & Sons, Ltd. ISBN 0-470-84726-3

[2] This chapter is based on J. P. Woodard, L. Hanzo: Comparative Study of Turbo Decoding Techniques: An Overview; IEEE Transactions on Vehicular Technology, Nov. 2000, Vol. 49, No. 6, pp 2208-2234 ©IEEE.

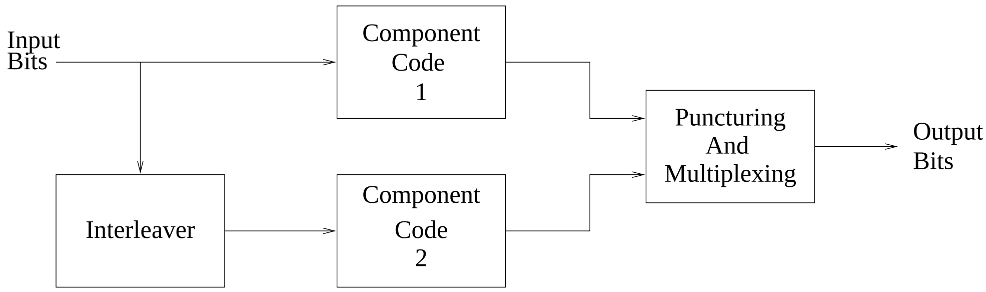


Figure 3.1: Turbo encoder schematic Berrou *et al.* ©IEEE, 1993 [12, 13].

second decoder in the first iteration is used as the a-priori information for the first decoder, and using this a-priori information the decoder can hopefully decode more bits correctly than it did in the first iteration. This cycle continues, with at each iteration both RSC decoders producing a soft output and extrinsic information based on the channel inputs and a-priori information obtained from the extrinsic information provided by the previous decoder. After each iteration the Bit Error Rate (BER) in the decoded sequence drops, but the improvements obtained with each iteration fall as the number of iterations increases so that for complexity reasons usually only between 4 and 12 iterations are used.

In their pioneering proposal Berrou, Glavieux and Thitimajashima [12] invoked a modified version of the classic minimum BER Maximum *A-Posteriori* (MAP) algorithm due to Bahl *et al.* [11] in the above iterative structure for decoding the constituent codes. Since the conception of turbo codes a large amount of work has been carried out in the area, aiming for example to reduce the decoder complexity, as suggested by Robertson, Villebrun and Höher [52] as well as by Berrou *et al.* [146]. Le Goff, Glavieux and Berrou [55], Wachsmann and Huber [56] as well as Robertson and Wörz [57] suggested using the codes in conjunction with bandwidth-efficient modulation schemes. Further advances in understanding the excellent preformance of the codes are due, for example, to Benedetto and Montorsi [58, 59] as well as to Perez, Seghers and Costello [60]. A number of seminal contributors including Hagenauer, Offer and Papke [61], as well as Pyndiah [147], extended the turbo concept to parallel concatenated block codes. Jung and Nasshan [65] characterised the coded performance under the constraints of short transmission frame length, which is characteristic of speech systems. In collaboration with Blanz they also applied turbo codes to a CDMA system using joint detection and antenna diversity [67]. Barbulescu and Pietrobon [68] as well as a number of other authors addressed the equally important issues of interlever design. Because of space limitations here we have to curtail listing the range of further contributors to the field, without whose advances this treatise could not have been written. It is particulary important to note the tutorial paper authored by Sklar [69].

Here we embark on describing turbo codes in more detail. Specifically, in Section 3.2 we detail the encoder used, while in Section 3.3 the decoder is portrayed. Then in Section 3.4 we characterise the performance of various turbo codes over Gaussian channels using BPSK. Then in Section 3.5 we discuss the employment of turbo codes over Rayleigh channels, and characterise the system's speech performance when using the G.729 speech codec.

3.2 Turbo Encoder

The general structure used in turbo encoders is shown in Figure 3.1. Two component codes are used to code the same input bits, but an interleaver is placed between the encoders. Generally RSC codes are used as the component codes, but it is possible to achieve good performance using a structure like that seen in Figure 3.1 with the aid of other component codes, such as for example block codes, as advocated by Hagenauer and Offer and Papke [61] as well as by Pyndiah [147]. Furthermore, it is also possible to employ more than two component codes. However, in this chapter we concentrate entirely on the standard turbo encoder structure

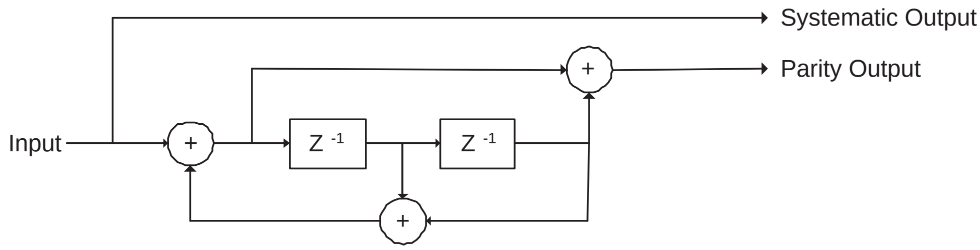


Figure 3.2: RSC encoder.

using two RSC codes. Turbo codes using block codes as component codes are described in Chapter 4.

The outputs from the two component codes are then punctured and multiplexed. Usually both component RSC codes are half rate, giving one parity bit and one systematic bit output for every input bit. Then to give an overall coding rate of one-half, half the output bits from the two encoders must be punctured. The arrangement that is often favoured, and that we have used in our work, is to transmit all the systematic bits from the first RSC encoder, and half the parity bits from each encoder. Note that the systematic bits are rarely punctured, since this degrades the performance of the code more dramatically than puncturing the parity bits.

Figure 3.2 shows the K=3 RSC code we have used as the component codes in most of our simulations. This code has generator polynomials 7 (for the feedback polynomial) and 5.

Parallel concatenated codes had been well investigated before Berrou *et al.*'s breakthrough 1993 paper [12], but the dramatic improvement in performance that turbo codes gave arose because of the interleaver used between the encoders, and because recursive codes were used as the component codes. Recently theoretical papers have been published for example by Benedetto and Montorsi [58, 59] which endeavour to explain the remarkable performance of turbo codes. It appears that turbo codes can be thought of as having a performance gain proportional to the interleaver length used. However, the decoding complexity per bit does not depend on the interleaver length. Therefore extremely good performance can be achieved with reasonable complexity by using very long interleavers. However, for many important applications, such as speech transmission, extremely long frame lengths are not practical because of the delays they result in. Therefore in this chapter we have also investigated the use of turbo codes in conjunction with short frame lengths of the order of 100 bits.

3.3 Turbo Decoder

3.3.1 Introduction

The general structure of an iterative turbo decoder is shown in Figure 3.3. Two component decoders are linked by interleavers in a structure similar to that of the encoder. As seen in the figure, each decoder takes three inputs–: the systematically encoded channel output bits, the parity bits transmitted from the associated component encoder, and the information from the other component decoder about the likely values of the bits concerned. This information from the other decoder is referred to as a-priori information. The component decoders have to exploit both the inputs from the channel and this a-priori information. They must also provide what are known as soft outputs for the decoded bits. This means that as well as providing the decoded output bit sequence, the component decoders must also give the associated probabilities for each bit that has been correctly decoded. The soft outputs are typically represented in terms of the so-called Log Likelihood Ratios (LLRs). The polarity of the LLR determines the sign of the bit, while its amplitude quantifies the probability of a correct decision. These LLRs are described in Section 3.3.2. Two suitable decoders are the Soft-Output Viterbi Algorithm (SOVA) proposed by Hagenauer and Höher [53] and Bahl's Maximum A-Posteriori (MAP) [11] algorithm, which are described in Sections 3.3.6 and 3.3.3,

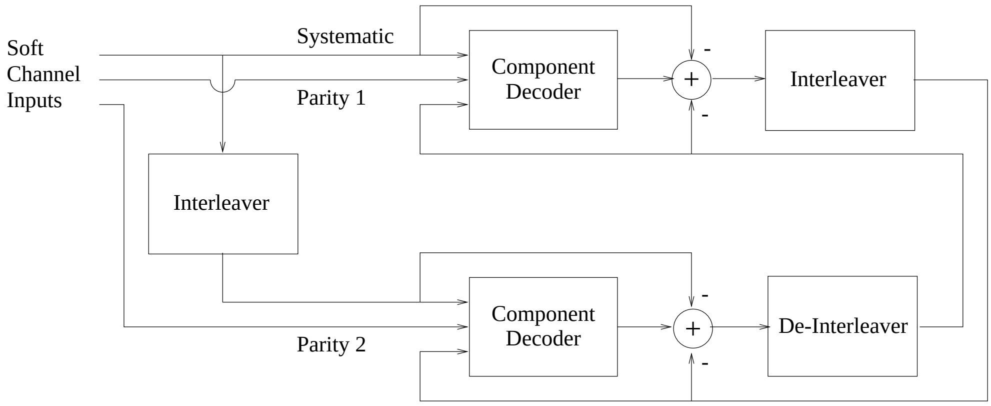


Figure 3.3: Turbo decoder schematic Berrou *et al.* ©IEEE, 1993 [12, 13].

respectively.

The decoder of Figure 3.3 operates iteratively, and in the first iteration the first component decoder takes channel output values only, and produces a soft output as its estimate of the data bits. The soft output from the first encoder is then used as additional information for the second decoder, which uses this information along with the channel outputs to calculate its estimate of the data bits. Now the second iteration can begin, and the first decoder decodes the channel outputs again, but now with additional information about the value of the input bits provided by the output of the second decoder in the first iteration. This additional information allows the first decoder to obtain a more accurate set of soft outputs, which are then used by the second decoder as a-priori information. This cycle is repeated, and with every iteration the BER of the decoded bits tends to fall. However, the improvement in performance obtained with increasing numbers of iterations decreases as the number of iterations increases. Hence, for complexity reasons, usually only about eight iterations are used.

Owing to the interleaving used at the encoder, care must be taken to properly interleave and de-interleave the LLRs which are used to represent the soft values of the bits, as seen in Figure 3.3. Furthermore, because of the iterative nature of the decoding, care must be taken not to reuse the same information more than once at each decoding step. For this reason the concept of so-called extrinsic and intrinsic information was used in their seminal paper by Berrou *et al.* describing iterative decoding of turbo codes [12]. These concepts and the reason for the subtraction circles shown in Figure 3.3 are described in Section 3.3.4.

Other, non-iterative, decoders have been proposed [148, 149] which give optimal decoding of turbo codes. However, the improvement in performance over iterative decoders was found to be only about 0.35 dB, and they are hugely complex. Therefore the iterative scheme shown in Figure 3.3 is commonly used. We now proceed with describing the concepts and algorithms used in the iterative decoding of turbo codes, commencing with the portrayal of LLRs.

3.3.2 Log Likelihood Ratios

The concept of LLRs was shown by Robertson [150] to simplify the passing of information from one component decoder to the other in the iterative decoding of turbo codes, and so is now widely used in the turbo coding literature. The LLR of a data bit u_k is denoted as $L(u_k)$ and is defined to be merely the log of the ratio of the probabilities of the bit taking its two possible values, i.e.:

$$L(u_k) \triangleq \ln\left(\frac{P(u_k = +1)}{P(u_k = -1)}\right). \tag{3.1}$$

Notice that the two possible values for the bit u_k are taken to be $+1$ and -1, rather than 1 and 0. This definition of the two values of a binary variable makes no conceptual difference, but it slightly simplifies

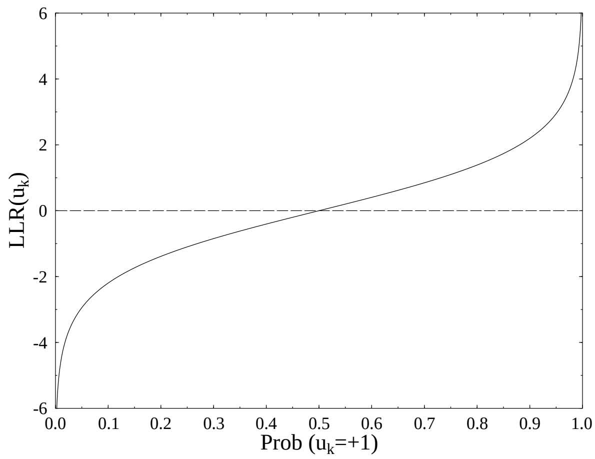


Figure 3.4: LLR $L(u_k)$ versus the probability of $u_k = +1$.

the mathematics in the derivations which follow. Hence this convention is used throughout this chanpter. Figure 3.4 shows how $L(u_k)$ varies as the probability of $u_k = +1$ varies. It can be seen from this figure that the sign of the LLR $L(u_k)$ of a bit u_k will indicate whether the bit is more likely to be $+1$ or -1, and the magnitude of the LLR gives an indication of how likely it is that the sign of the LLR gives the correct value of u_k. When the LLR $L(u_k) \approx 0$, we have $P(u_k = +1) \approx P(u_k = -1) \approx 0.5$, and we cannot be certain about the value of u_k. Conversely, when $L(u_k) \gg 0$, we have $P(u_k = +1) \gg P(u_k = -1)$ and we can be almost certain that $u_k = +1$.

Given the LLR $L(u_k)$, it is possible to calculate the probability that $u_k = +1$ or $u_k = -1$ as follows. Remembering that $P(u_k = -1) = 1 - P(u_k = +1)$, and taking the exponent of both sides in Equation 3.1, we can write:

$$\mathrm{e}^{L(u_k)} = \frac{P(u_k = +1)}{1 - P(u_k = +1)}, \tag{3.2}$$

so:

$$\begin{aligned} P(u_k = +1) &= \frac{\mathrm{e}^{L(u_k)}}{1 + \mathrm{e}^{L(u_k)}} \\ &= \frac{1}{1 + \mathrm{e}^{-L(u_k)}}. \end{aligned} \tag{3.3}$$

Similarly:

$$\begin{aligned} P(u_k = -1) &= \frac{1}{1 + \mathrm{e}^{+L(u_k)}} \\ &= \frac{\mathrm{e}^{-L(u_k)}}{1 + \mathrm{e}^{-L(u_k)}}, \end{aligned} \tag{3.4}$$

and hence we can write:

$$P(u_k = \pm 1) = \left(\frac{\mathrm{e}^{-L(u_k)/2}}{1 + \mathrm{e}^{-L(u_k)}} \right) \cdot \mathrm{e}^{\pm L(u_k)/2}. \tag{3.5}$$

Notice that the bracketed term in this equation does not depend on whether we are interested in the probability that $u_k = +1$ or -1, and so it can be treated as a constant in certain applications, such as in Section 3.3.3 where we use this equation in the derivation of the MAP algorithm.

As well as the LLR $L(u_k)$ based on the unconditional probabilities $P(u_k = \pm 1)$, we are also interested in LLRs based on conditional probabilities. For example, in channel coding theory we are interested in the probability that $u_k = \pm 1$ based, or conditioned, on some received sequence $\underline{y}$, and hence we may use the conditional LLR $L(u_k|\underline{y})$, which is defined as:

$$L(u_k|\underline{y}) \triangleq \ln\left(\frac{P(u_k = +1|\underline{y})}{P(u_k = -1|\underline{y})}\right). \tag{3.6}$$

The conditional probabilities $P(u_k = \pm 1|\underline{y})$ are known as the a-posteriori probabilities of the decoded bit u_k, and it is these a-posteriori probabilities that our soft-in soft-out decoders described in later sections attempt to find.

Apart from the conditional LLR $L(u_k|\underline{y})$ based on the a-posteriori probabilities $P(u_k = \pm 1|\underline{y})$, we will also use conditional LLRs based on the probability that the receiver's matched filter output would be y_k given that the corresponding transmitted bit x_k was either $+1$ or -1. This conditional LLR is written as $L(y_k|x_k)$ and is defined as:

$$L(y_k|x_k) \triangleq \ln\left(\frac{P(y_k|x_k = +1)}{P(y_k|x_k = -1)}\right). \tag{3.7}$$

Notice the conceptual difference between the definitions of $L(u_k|\underline{y})$ in Equation 3.6 and $L(y_k|x_k)$ in Equation 3.7, despite these two conditional LLRs being represented with very similar notation. This contrast in the definitions of conditional LLRs is somewhat confusing, but since these definitions are widely used in the turbo coding literature, we have introduced them here.

If we assume that the transmitted bit $x_k = \pm 1$ has been sent over a Gaussian or fading channel using BPSK modulation, then we can write for the probability of the matched filter output y_k that:

$$P(y_k|x_k = +1) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{E_b}{2\sigma^2}(y_k - a)^2\right), \tag{3.8}$$

where E_b is the transmitted energy per bit, σ^2 is the noise variance and a is the fading amplitude (we have $a = 1$ for non-fading AWGN channels). Similarly, we have:

$$P(y_k|x_k = -1) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{E_b}{2\sigma^2}(y_k + a)^2\right). \tag{3.9}$$

Therefore, when we use BPSK over a (possibly fading) Gaussian channel, we can rewrite Equation 3.7 as:

$$\begin{aligned}
L(y_k|x_k) &\triangleq \ln\left(\frac{P(y_k|x_k = +1)}{P(y_k|x_k = -1)}\right) \\
&= \ln\left(\frac{\exp\left(-\frac{E_b}{2\sigma^2}(y_k - a)^2\right)}{\exp\left(-\frac{E_b}{2\sigma^2}(y_k + a)^2\right)}\right) \\
&= \left(-\frac{E_b}{2\sigma^2}(y_k - a)^2\right) - \left(-\frac{E_b}{2\sigma^2}(y_k + a)^2\right) \\
&= \frac{E_b}{2\sigma^2} 4a \cdot y_k \\
&= L_c y_k,
\end{aligned} \tag{3.10}$$

where:

$$L_c = 4a\frac{E_b}{2\sigma^2} \tag{3.11}$$

is defined as the channel reliability value, and depends only on the signal-to-noise ratio (SNR) and fading amplitude of the channel. Hence, for BPSK over a (possibly fading) Gaussian channel, the conditional LLR $L(y_k|x_k)$, which is referred to as the soft output of the channel, is simply the matched filter output y_k multiplied by the channel reliability value L_c.

Having introduced LLRs, we now proceed to describe the operation of the MAP algorithm, which is one of the possible soft-in soft-out component decoders that can be used in an iterative turbo decoder.

3.3.3 The Maximum A-Posteriori Algorithm

3.3.3.1 Introduction and Mathematical Preliminaries

In 1974 an algorithm, known as the Maximum A-Posteriori (MAP) algorithm, was proposed by Bahl, Cocke, Jelinek and Raviv for estimating the a-posteriori probabilities of the states and the transitions of an observed Markov source, when subjected to memoryless noise. This algorithm has also become known as the BCJR algorithm, named after its inventors. They showed how the algorithm could be used for decoding both block and convolutional codes. When employed for decoding convolutional codes, the algorithm is optimal in terms of minimising the decoded, unlike the Viterbi algorithm [9], which minimises the probability of an incorrect path through the trellis being selected by the decoder. Thus the Viterbi algorithm can be thought of as minimising the number of *groups* of bits associated with these trellis paths, rather than the actual number of bits, which are decoded incorrectly. Nevertheless, as stated by Bahl *et al.* in [11], in most applications the performance of the two algorithms will be almost identical. However, the MAP algorithm examines every possible path through the convolutional decoder trellis and therefore initially seemed to be unfeasibly complex for application in most systems. Hence it was not widely used before the discovery of turbo codes.

However, the MAP algorithm provides not only the estimated bit sequence, but also the probabilities for each bit that has been decoded correctly. This is essential for the iterative decoding of turbo codes proposed by Berrou *et al.* [12], and so MAP decoding was used in this seminal paper. Since then much work has been done to reduce the complexity of the MAP algorithm to a reasonable level. In this section we describe the theory behind the MAP algorithm as used for the soft-output decoding of the component convolutional codes of turbo codes. Throughout our work it is assumed that binary codes are used.

We use Bayes' rule repeatedly throughout this section. This rule gives the joint probability of a *and* b, $P(a \wedge b)$, in terms of the conditional probability of a *given* b as:

$$P(a \wedge b) = P(a|b) \cdot P(b). \tag{3.12}$$

A useful consequence of Bayes' rule is that:

$$P(\{a \wedge b\}|c) = P(a|\{b \wedge c\}) \cdot P(b|c), \tag{3.13}$$

which can be derived from Equation 3.12 by considering $x \equiv a \wedge b$ and $y \equiv b \wedge c$ as follows. From Equation 3.12 we can write:

$$\begin{aligned} P(\{a \wedge b\}|c) &\equiv P(x|c) = \frac{P(x \wedge c)}{P(c)} \\ &= \frac{P(a \wedge b \wedge c)}{P(c)} \equiv \frac{P(a \wedge y)}{P(c)} \\ &= \frac{P(a|y) \cdot P(y)}{P(c)} \equiv P(a|\{b \wedge c\}) \cdot \frac{P(b \wedge c)}{P(c)} \\ &= P(a|\{b \wedge c\}) \cdot P(b|c). \end{aligned} \tag{3.14}$$

The MAP algorithm gives, for each decoded bit u_k, the probability that this bit was $+1$ or -1, given the received symbol sequence $\underline{y}$. As explained in Section 3.3.2 this is equivalent to finding the a-posteriori LLR $L(u_k|\underline{y})$, where:

$$L(u_k|\underline{y}) = \ln\left(\frac{P(u_k = +1|\underline{y})}{P(u_k = -1|\underline{y})}\right). \tag{3.15}$$

Bayes' rule allows us to rewrite this equation as:

$$L(u_k|\underline{y}) = \ln\left(\frac{P(u_k = +1 \wedge \underline{y})}{P(u_k = -1 \wedge \underline{y})}\right). \tag{3.16}$$

Let us now consider Figure 3.5 showing the transitions possible for the $K = 3$ RSC code shown in

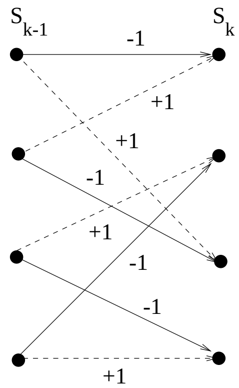


Figure 3.5: Possible transitions in K=3 RSC component code.

Figure 3.2, which we have used for the component codes in most of our work. For this $K = 3$ code there are four encoder states, and since we consider a binary code, in each encoder state two state transitions are possible, depending on the value of this bit. One of these transitions is associated with the input bit of -1 shown as a continuous line, while the other transition corresponds to the input bit of $+1$ shown as a broken line. It can be seen from Figure 3.5 that if the previous state S_{k-1} and the present state S_k are known, then the value of the input bit u_k, which caused the transition between these two states, will be known. Hence the probability that $u_k = +1$ is equal to the probability that the transition from the previous state S_{k-1} to the present state S_k is one of the set of four possible transitions that can occur when $u_k = +1$ (i.e. those transitions shown with broken lines). This set of transitions is mutually exclusive (i.e. only one of them could have occured at the encoder), and so the probability that any one of them occurs is equal to the sum of their individual probabilities. Hence we can rewrite Equation 3.16 as:

$$L(u_k|\underline{y}) = \ln \left(\frac{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=+1}} P(S_{k-1} = \grave{s} \wedge S_k = s \wedge \underline{y})}{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=-1}} P(S_{k-1} = \grave{s} \wedge S_k = s \wedge \underline{y})} \right), \tag{3.17}$$

where $(\grave{s}, s) \Rightarrow u_k = +1$ is the set of transitions from the previous state $S_{k-1} = \grave{s}$ to the present state $S_k = s$ that can occur if the input bit $u_k = +1$, and similarly for $(\grave{s}, s) \Rightarrow u_k = -1$. For brevity we shall write $P(S_{k-1} = \grave{s} \wedge S_k = s \wedge \underline{y})$ as $P(\grave{s} \wedge s \wedge \underline{y})$.

We now consider the individual probabilities $P(\grave{s} \wedge s \wedge \underline{y})$ from the numerator and denominator of Equation 3.17. The received sequence $\underline{y}$ can be split up into three sections: the received codeword associated with the present transition $\underline{y}_k$, the received sequence prior to the present transition $\underline{y}_{j<k}$ and the received sequence after the present transition $\underline{y}_{j>k}$. This split is shown in Figure 3.6, again for the example of our $K = 3$ RSC component code shown in Figure 3.2. We can thus write for the individual probabilities $P(\grave{s} \wedge s \wedge \underline{y})$:

$$P(\grave{s} \wedge s \wedge \underline{y}) = P(\grave{s} \wedge s \wedge \underline{y}_{j<k} \wedge \underline{y}_k \wedge \underline{y}_{j>k}). \tag{3.18}$$

Using Bayes' rule of $P(a \wedge b) = P(a|b)P(b)$ and the fact that if we assume that the channel is memoryless, then the future received sequence $\underline{y}_{j>k}$ will depend only on the present state s and not on the previous state $\grave{s}$ or the present and previous received channel sequences $\underline{y}_k$ and $\underline{y}_{j<k}$, we can write:

$$\begin{aligned} P(\grave{s} \wedge s \wedge \underline{y}) &= P(\underline{y}_{j>k}|\{\grave{s} \wedge s \wedge \underline{y}_{j<k} \wedge \underline{y}_k\}) \cdot P(\grave{s} \wedge s \wedge \underline{y}_{j<k} \wedge \underline{y}_k) \\ &= P(\underline{y}_{j>k}|s) \cdot P(\grave{s} \wedge s \wedge \underline{y}_{j<k} \wedge \underline{y}_k). \end{aligned} \tag{3.19}$$

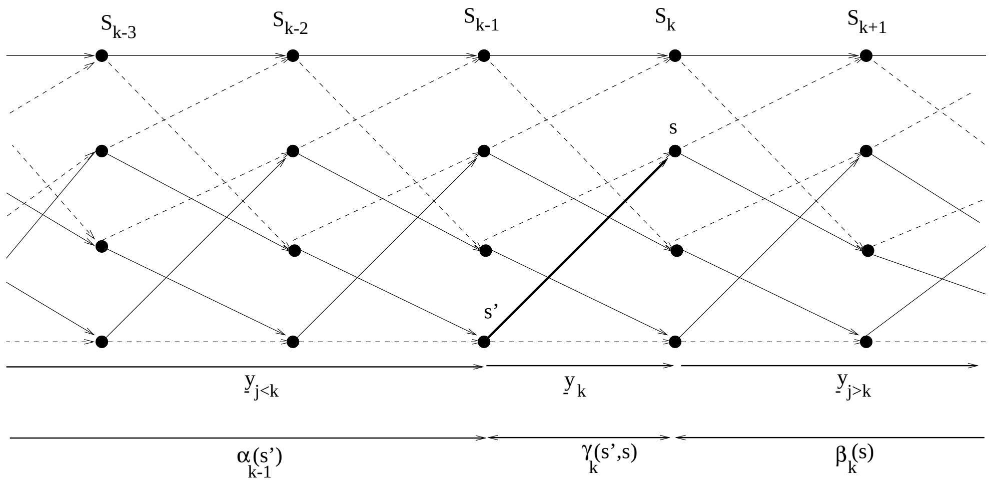


Figure 3.6: MAP decoder trellis for $K = 3$ RSC code.

Again, using Bayes' rule and the assumption that the channel is memoryless, we can expand Equation 3.19 as follows:

$$\begin{aligned} P(\grave{s} \wedge s \wedge \underline{y}) &= P(\underline{y}_{j>k}|s) \cdot P(\grave{s} \wedge s \wedge \underline{y}_{j<k} \wedge \underline{y}_k) \\ &= P(\underline{y}_{j>k}|s) \cdot P(\{\underline{y}_k \wedge s\}|\{\grave{s} \wedge \underline{y}_{j<k}\}) \cdot P(\grave{s} \wedge \underline{y}_{j<k}) \\ &= P(\underline{y}_{j>k}|s) \cdot P(\{\underline{y}_k \wedge s\}|\grave{s}) \cdot P(\grave{s} \wedge \underline{y}_{j<k}) \\ &= \beta_k(s) \cdot \gamma_k(\grave{s}, s) \cdot \alpha_{k-1}(\grave{s}), \end{aligned} \tag{3.20}$$

where:

$$\alpha_{k-1}(\grave{s}) = P(S_{k-1} = \grave{s} \wedge \underline{y}_{j<k}) \tag{3.21}$$

is the probability that the trellis is in state $\grave{s}$ at time $k-1$ and the received channel sequence up to this point is $\underline{y}_{j<k}$, as visualised in Figure 3.6:

$$\beta_k(s) = P(\underline{y}_{j>k}|S_k = s) \tag{3.22}$$

is the probability that given the trellis is in state s at time k the future received channel sequence will be $\underline{y}_{j>k}$, and lastly:

$$\gamma_k(\grave{s}, s) = P(\{\underline{y}_k \wedge S_k = s\}|S_{k-1} = \grave{s}) \tag{3.23}$$

is the probability that given the trellis was in state $\grave{s}$ at time $k-1$, it moves to state s and the received channel sequence for this transition is $\underline{y}_k$.

Equation 3.20 shows that the probability $P(\grave{s} \wedge s \wedge \underline{y})$ that the encoder trellis took the transition from state $S_{k-1} = \grave{s}$ to state $S_k = s$ and the received sequence is $\underline{y}$, can be split into the product of three terms: $\alpha_{k-1}(\grave{s})$, $\gamma_k(\grave{s}, s)$ and $\beta_k(s)$. The meaning of these three probability terms is shown in Figure 3.6, for the transition $S_{k-1} = \grave{s}$ to $S_k = s$ shown by the bold line in this figure. The MAP algorithm finds $\alpha_k(s)$ and $\beta_k(s)$ for all states s throughout the trellis, i.e. for $k = 0, 1, \ldots, N-1$, and $\gamma_k(\grave{s}, s)$ for all possible transitions from state $S_{k-1} = \grave{s}$ to state $S_k = s$, again for $k = 0, 1 \ldots, N-1$. These values are then used to find the probabilities $P(S_{k-1} = \grave{s} \wedge S_k = s \wedge \underline{y})$ of Equation 3.20, which are then used in Equation 3.17 to give the LLRs $L(u_k|\underline{y})$ for each bit u_k. These operations are summarised in the flowchart of Figure 3.8 below. We now describe how the values $\alpha_k(s)$, $\beta_k(s)$ and $\gamma_k(\grave{s}, s)$ can be calculated.

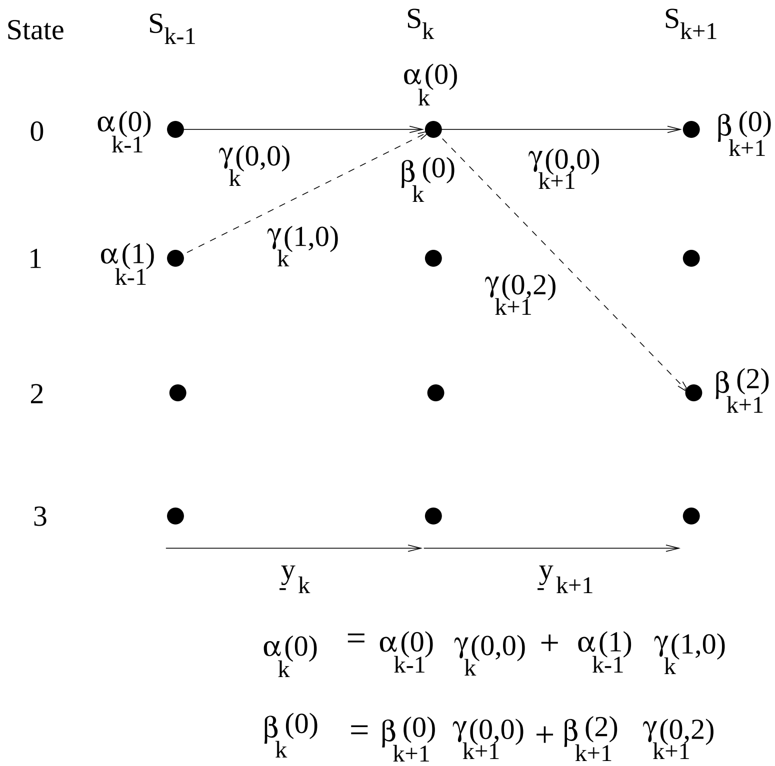


Figure 3.7: Recursive calculation of $\alpha_k(0)$ and $\beta_k(0)$.

3.3.3.2 Forward Recursive Calculation of the $\alpha_k(s)$ Values

Consider first $\alpha_k(s)$. From the definition of $\alpha_{k-1}(\grave{s})$ in Equation 3.21 we can write:

$$\begin{aligned} \alpha_k(s) &= P(S_k = s \wedge \underline{y}_{j<k+1}) \\ &= P(s \wedge \underline{y}_{j<k} \wedge \underline{y}_k) \\ &= \sum_{\text{all } \grave{s}} P(s \wedge \grave{s} \wedge \underline{y}_{j<k} \wedge \underline{y}_k), \end{aligned} \tag{3.24}$$

where in the last line we split the probability $P(s \wedge \underline{y}_{y<k+1})$ into the sum of joint probabilities $P(s \wedge \grave{s} \wedge \underline{y}_{j<k+1})$ over all possible previous states $\grave{s}$. Using Bayes' rule and the assumption that the channel is memoryless again, we can proceed as follows:

$$\begin{aligned} \alpha_k(s) &= \sum_{\text{all } \grave{s}} P(s \wedge \grave{s} \wedge \underline{y}_{j<k} \wedge \underline{y}_k) \\ &= \sum_{\text{all } \grave{s}} P(\{s \wedge \underline{y}_k\} | \{\grave{s} \wedge \underline{y}_{j<k}\}) \cdot P(\grave{s} \wedge \underline{y}_{j<k}) \\ &= \sum_{\text{all } \grave{s}} P(\{s \wedge \underline{y}_k\} | \grave{s}) \cdot P(\grave{s} \wedge \underline{y}_{j<k}) \\ &= \sum_{\text{all } \grave{s}} \gamma_k(\grave{s}, s) \cdot \alpha_{k-1}(\grave{s}). \end{aligned} \tag{3.25}$$

Thus, once the $\gamma_k(\grave{s}, s)$ values are known, the $\alpha_k(s)$ values can be calculated recursively. Assuming that the trellis has the initial state $S_0 = 0$, the initial conditions for this recursion are:

$$\begin{aligned} \alpha_0(S_0 = 0) &= 1 \\ \alpha_0(S_0 = s) &= 0 \quad \text{for all } s \neq 0. \end{aligned} \tag{3.26}$$

Figure 3.7 shows an example of how one $\alpha_k(s)$ value, for $s = 0$, is calculated recursively using values of $\alpha_{k-1}(\grave{s})$ and $\gamma_k(\grave{s}, s)$ for our example $K = 3$ RSC code. Notice that, as we are considering a binary trellis, only two previous states, $S_{k=1} = 0$ and $S_{k-1} = 1$, have paths to the state $S_k = 0$. Therefore $\gamma_k(\grave{s}, s)$ will be non-zero only for $\grave{s} = 0$ or $\grave{s} = 1$ and hence the summation in Equation 3.25 is over only two terms.

3.3.3.3 Backward Recursive Calculation of the $\beta_k(s)$ Values

The values of $\beta_k(s)$ can similarly be calculated recursively as shown below. From the definition of $\beta_k(s)$ in Equation 3.22, we can write $\beta_{k-1}(\grave{s})$ as:

$$\beta_{k-1}(\grave{s}) = P(\underline{y}_{j>k-1} | S_{k-1} = \grave{s}), \tag{3.27}$$

and again splitting a single probability into the sum of joint probabilities and using the derivation from Bayes' rule in Equation 3.13, as well as the assumption that the channel is memoryless, we have:

$$\begin{aligned} \beta_{k-1}(\grave{s}) &= P(\underline{y}_{j>k-1} | \grave{s}) \\ &= \sum_{\text{all } s} P(\{\underline{y}_{j>k-1} \wedge s\} | \grave{s}) \\ &= \sum_{\text{all } s} P(\{\underline{y}_k \wedge \underline{y}_{j>k} \wedge s\} | \grave{s}) \\ &= \sum_{\text{all } s} P(\underline{y}_{j>k} | \{\grave{s} \wedge s \wedge \underline{y}_k\}) \cdot P(\{\underline{y}_k \wedge s\} | \grave{s}) \\ &= \sum_{\text{all } s} P(\underline{y}_{j>k} | s) \cdot P(\{\underline{y}_k \wedge s\} | \grave{s}) \\ &= \sum_{\text{all } s} \beta_k(s) \cdot \gamma_k(\grave{s}, s). \end{aligned} \tag{3.28}$$

Thus, once the values $\gamma_k(\grave{s}, s)$ are known, a backward recursion can be used to calculate the values of $\beta_{k-1}(\grave{s})$ from the values of $\beta_k(s)$ using Equation 3.28. Figure 3.7 again shows an example of how the $\beta_k(0)$ value is calculated recursively using values of $\beta_{k+1}(s)$ and $\gamma_{k+1}(0, s)$ for our example $K = 3$ RSC code.

The initial conditions which should be used for $\beta_N(s)$ are not as clear as for $\alpha_0(s)$. From Equation 3.22 $\beta_k(s)$ is the probability that the future received sequence is $\underline{y}_{j>k}$, given that the present state is s. For the last stage in the trellis, however, i.e. when $k = N$, there is no future received sequence, and hence it is not clear what the initial values $\beta_N(s)$ should be set to. Berrou *et al.* [12] used the initial values $\beta_N(0) = 1$ and $\beta_N(s) = 0$ for all $s \neq 0$ for a trellis terminated in the all-zero state, and in [150] the initial conditions for an unterminated trellis were given by Robertson as $\beta_N(s) = \alpha_N(s)$ for all s. However, as pointed out by Breiling [151], if we consider $\beta_{N-1}(s)$ from Equation 3.22 we have:

$$\begin{aligned} \beta_{N-1}(\grave{s}) &= P(\underline{y}_N | \grave{s}) \\ &= \sum_{\text{all } s} P(\{\underline{y}_N \wedge s\} | \grave{s}) \\ &= \sum_{\text{all } s} \gamma_N(\grave{s}, s) \end{aligned} \tag{3.29}$$

and from the backward recursion for $\beta_{k-1}(\grave{s})$ in Equation 3.28 we have:

$$\beta_{N-1}(\grave{s}) = \sum_{\text{all } s} \beta_N(s) \gamma_N(\grave{s}, s). \tag{3.30}$$

For both Equation 3.29 and Equation 3.30 to be satisfied, we must have:

$$\beta_N(s) = 1 \quad \text{for all } s. \tag{3.31}$$

If the trellis is terminated, so that only the final state $S_N = 0$ is possible, this can be taken into account in the backward recursive calculation of the $\beta_k(s)$ values through the $\gamma_k(\grave{s}, s)$ values. In a terminated trellis for the last $K-1$ transitions, where K is the constraint length of the convolutional code, for each state s only one transition (the one which takes the trellis towards the all-zero state) will be possible. Hence $\gamma_k(\grave{s}, s) = P(\{\underline{y}_k \wedge S_k = s\}|S_{k-1} = \grave{s})$ will be zero for all values of s except one, and with the initial values $\beta_N(s) = 1$ the correct values of $\beta_{N-1}(s), \beta_{N-2}(s), \ldots, \beta_{N-K+1}$ will be calculated through Equation 3.28. Thus theory indicates that we should use $\beta_N(s) = 1$ for all s and account for the trellis termination by setting the values of $\gamma_k(\grave{s}, s)$ to zero for all transitions that are not possible owing to trellis termination. However, the same result can be achieved using β values of $\beta_N(0) = 1$ and $\beta_N(s) = 0$ for $s \neq 0$, as suggested by Berrou *et al*. [12], and calculating $\gamma_k(\grave{s}, s)$ values in the same way as for all other transitions (i.e. directly from the channel inputs — see next section). This second method is simpler to implement and hence it is more commonly used in practice.

3.3.3.4 Calculation of the $\gamma_k(\grave{s}, s)$ Values

We now consider how the $\gamma_k(\grave{s}, s)$ values in Equation 3.20 can be calculated from the received channel sequence. Using the definition of $\gamma_k(\grave{s}, s)$ from Equation 3.23 and the derivation from Bayes' rule given in Equation 3.13 we have:

$$\begin{aligned}
\gamma_k(\grave{s}, s) &= P(\{\underline{y}_k \wedge s\}|\grave{s}) \\
&= P(\underline{y}_k|\{\grave{s} \wedge s\}) \cdot P(s|\grave{s}) \\
&= P(\underline{y}_k|\{\grave{s} \wedge s\}) \cdot P(u_k),
\end{aligned} \tag{3.32}$$

where u_k is the input bit necessary to cause the transition from state $S_{k-1} = \grave{s}$ to state $S_k = s$, and $P(u_k)$ is the a-priori probability of this bit. From Equation 3.5 this can be written as:

$$\begin{aligned}
P(u_k) &= \left(\frac{\mathrm{e}^{-L(u_k)/2}}{1 + \mathrm{e}^{-L(u_k)}}\right) \cdot \mathrm{e}^{(u_k L(u_k)/2)} \\
&= C^{(1)}_{L(u_k)} \cdot \mathrm{e}^{(u_k L(u_k)/2)},
\end{aligned} \tag{3.33}$$

where, as stated before:

$$C^{(1)}_{L(u_k)} = \left(\frac{\mathrm{e}^{-L(u_k)/2}}{1 + \mathrm{e}^{-L(u_k)}}\right) \tag{3.34}$$

depends only on the LLR $L(u_k)$ and not on whether u_k is $+1$ or -1.

The first term in the second and third lines of Equation 3.32, $P(\underline{y}_k|\{\grave{s} \wedge s\})$, is equivalent to $P(\underline{y}_k|\underline{x}_k)$, where $\underline{x}_k$ is the transmitted codeword associated with the transition from state $S_{k-1} = \grave{s}$ to state $S_k = s$. Again assuming the channel is memoryless we can write:

$$P(\underline{y}_k|\{\grave{s} \wedge s\}) \equiv P(\underline{y}_k|\underline{x}_k) = \prod_{l=1}^{n} P(y_{kl}|x_{kl}), \tag{3.35}$$

where x_{kl} and y_{kl} are the individual bits within the transmitted and received codewords $\underline{x}_k$ and $\underline{y}_k$, and n is the number of these bits in each codeword $\underline{y}_k$ or $\underline{x}_k$. Assuming that the transmitted bits x_{kl} have been transmitted over a Gaussian channel using BPSK, so that the transmitted symbols are either $+1$ or -1, we have for $P(y_{kl}|x_{kl})$:

$$P(y_{kl}|x_{kl}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{E_b}{2\sigma^2}(y_{kl} - a x_{kl})^2\right), \tag{3.36}$$

where E_b is the transmitted energy per bit, σ^2 is the noise variance and a is the fading amplitude (a=1 for non-fading AWGN channels). Upon substituting Equation 3.36 in Equation 3.35 we have:

$$\begin{aligned}
P(\underline{y}_k|\{\grave{s} \wedge s\}) &= \prod_{l=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{E_b}{2\sigma^2}(y_{kl} - ax_{kl})^2\right) \\
&= \frac{1}{(\sqrt{2\pi}\sigma)^n} \exp\left(-\frac{E_b}{2\sigma^2}\sum_{l=1}^{n}(y_{kl} - ax_{kl})^2\right) \\
&= \frac{1}{(\sqrt{2\pi}\sigma)^n} \exp\left(-\frac{E_b}{2\sigma^2}\sum_{l=1}^{n}(y_{kl}^2 + a^2x_{kl}^2 - 2ax_{kl}y_{kl})\right) \\
&= C_{\underline{y}_k}^{(2)} \cdot C_{\underline{x}_k}^{(3)} \cdot \exp\left(\frac{E_b}{2\sigma^2}2a\sum_{l=1}^{n} y_{kl}x_{yl}\right), \qquad (3.37)
\end{aligned}$$

where:

$$C_{\underline{y}_k}^{(2)} = \frac{1}{(\sqrt{2\pi}\sigma)^n} \cdot \exp\left(-\frac{E_b}{2\sigma^2}\sum_{l=1}^{n} y_{kl}^2\right) \qquad (3.38)$$

depends only on the channel SNR and on the magnitude of the received sequence $\underline{y}_k$, while:

$$\begin{aligned}
C_{\underline{x}_k}^{(3)} &= \exp\left(-\frac{E_b}{2\sigma^2}a^2\sum_{l=1}^{n} x_{kl}^2\right) \\
&= \exp\left(-\frac{E_b}{2\sigma^2}a^2 n\right) \qquad (3.39)
\end{aligned}$$

depends only on the channel SNR and on the fading amplitude. Hence we can write for $\gamma_k(\grave{s}, s)$:

$$\begin{aligned}
\gamma_k(\grave{s}, s) &= P(u_k) \cdot P(\underline{y}_k|\{\grave{s} \wedge s\}) \\
&= C \cdot \mathrm{e}^{(u_k L(u_k)/2)} \cdot \exp\left(\frac{E_b}{2\sigma^2}2a\sum_{l=1}^{n} y_{kl}x_{yl}\right) \\
&= C \cdot \mathrm{e}^{(u_k L(u_k)/2)} \cdot \exp\left(\frac{L_c}{2}\sum_{l=1}^{n} y_{kl}x_{yl}\right), \qquad (3.40)
\end{aligned}$$

where

$$C = C_{L(u_k)}^{(1)} \cdot C_{\underline{y}_k}^{(2)} \cdot C_{\underline{x}_k}^{(3)}. \qquad (3.41)$$

The term C does not depend on the sign of the bit u_k or the transmitted codeword $\underline{x}_k$ and so is constant over the summations in the numerator and denominator in Equation 3.17 and cancels out.

From Equations 3.17 and 3.20 we can write for the conditional LLR of u_k, given the received sequence $\underline{y}_k$,:

$$\begin{aligned}
L(u_k|\underline{y}) &= \ln\left(\frac{\sum_{\substack{(\grave{s},s)\Rightarrow \\ u_k=+1}} P(S_{k-1} = \grave{s} \wedge S_k = s \wedge \underline{y})}{\sum_{\substack{(\grave{s},s)\Rightarrow \\ u_k=-1}} P(S_{k-1} = \grave{s} \wedge S_k = s \wedge \underline{y})}\right) \\
&= \ln\left(\frac{\sum_{\substack{(\grave{s},s)\Rightarrow \\ u_k=+1}} \alpha_{k-1}(\grave{s}) \cdot \gamma_k(\grave{s}, s) \cdot \beta_k(s)}{\sum_{\substack{(\grave{s},s)\Rightarrow \\ u_k=-1}} \alpha_{k-1}(\grave{s}) \cdot \gamma_k(\grave{s}, s) \cdot \beta_k(s)}\right). \qquad (3.42)
\end{aligned}$$

It is this conditional LLR $L(u_k|\underline{y})$ that the MAP decoder delivers.

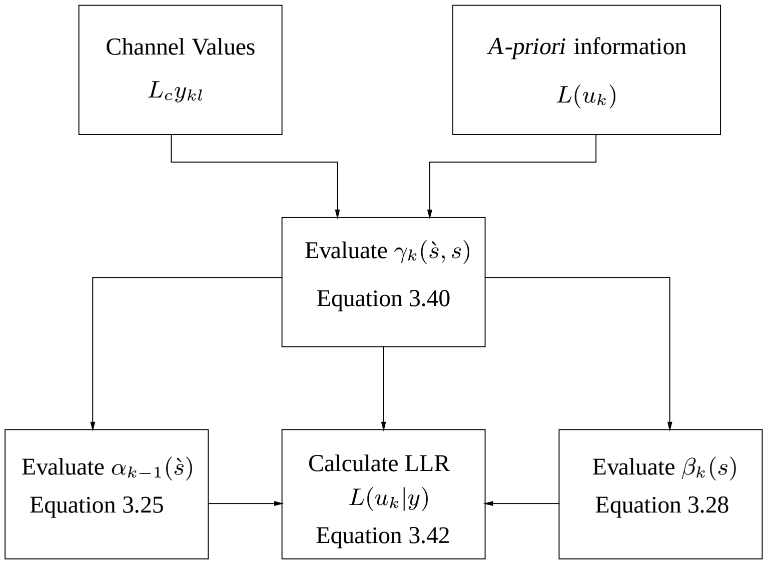


Figure 3.8: Summary of the key operations in the MAP algorithm.

3.3.3.5 Summary of the MAP Algorithm

From the description given above, we see that the MAP decoding of a received sequence $\underline{y}$ to give the a-posteriori LLR $L(u_k|\underline{y})$ can be carried out as follows. As the channel values y_{kl} are received, they and the a-priori LLRs $L(u_k)$ (which are provided in an iterative turbo decoder by the other component decoder — see Section 3.3.4) are used to calculate $\gamma_k(\grave{s}, s)$ according to Equation 3.40. The constant C can be omitted from the calculation of $\gamma_k(\grave{s}, s)$, as it will cancel out in the ratio in Equation 3.42. As the channel values y_{kl} are received, and the $\gamma_k(\grave{s}, s)$ values are calculated, the forward recursion from Equation 3.25 can be used to calculate $\alpha_k(\grave{s}, s)$. Once all the channel values have been received, and $\gamma_k(\grave{s}, s)$ has been calculated for all $k = 1, 2, \ldots, N$, the backward recursion from Equation 3.28 can be used to calculate the $\beta_k(\grave{s}, s)$ values. Finally, all the calculated values of $\alpha_k(\grave{s}, s)$, $\beta_k(\grave{s}, s)$ and $\gamma_k(\grave{s}, s)$ are used in Equation 3.42 to calculate the values of $L(u_k|\underline{y})$. These operations are summarised in the flowchart of Figure 3.8. Care must be taken to avoid numerical underflow problems in the recursive calculation of $\alpha_k(\grave{s}, s)$ and $\beta_k(\grave{s}, s)$, but such problems can be avoided by careful normalisation of these values. Such normalisation cancels out in the ratio in Equation 3.42 and so causes no change in the LLRs produced by the algorithm.

The MAP algorithm is, in the form described in this section, extremely complex owing to the multiplications needed in Equations 3.25 and 3.28 for the recursive calculation of $\alpha_k(\grave{s}, s)$ and $\beta_k(\grave{s}, s)$, the multiplications and exponential operations required to calculate $\gamma_k(\grave{s}, s)$ using Equation 3.40, and the multiplication and natural logarithm operations required to calculate $L(u_k|\underline{y})$ using Equation 3.42. However, much work has been done to reduce this complexity, and the Log-MAP algorithm [52], which will be described in Section 3.3.5, gives the same performance as the MAP algorithm but with a much lower complexity and without the numerical problems described above. We will first describe the principles behind the iterative decoding of turbo codes, and how the MAP algorithm described in this section can be used in such a scheme, before detailing the Log-MAP algorithm.

3.3.4 Iterative Turbo Decoding Principles

3.3.4.1 Turbo Decoding Mathematical Preliminaries

In this section we explain the concepts of extrinsic and intrinsic information as used by Berrou *et al.* [12], and highlight how the MAP algorithm described in the previous section, and other soft-in soft-out decoders, can be used in the iterative decoding of turbo codes.

Consider first the expression for $\gamma_k(\grave{s}, s)$ in Equation 3.40, which is restated here for convenience:

$$\gamma_k(\grave{s}, s) = C \cdot \mathrm{e}^{(u_k L(u_k)/2)} \cdot \exp\left(\frac{L_c}{2} \sum_{l=1}^{n} y_{kl} x_{yl}\right). \tag{3.43}$$

Since we are dealing with systematic codes, one of the n transmitted bits will be the systematic bit u_k. If we assume that this systematic bit is the first of the n transmitted bits, then we will have $x_{k1} = u_k$, and we can rewrite Equation 3.43 as:

$$\begin{aligned}\gamma_k(\grave{s}, s) &= C \cdot \mathrm{e}^{(u_k L(u_k)/2)} \cdot \exp\left(\frac{L_c}{2} y_{ks} u_k\right) \cdot \exp\left(\frac{L_c}{2} \sum_{l=2}^{n} y_{kl} x_{yl}\right) \\ &= C \cdot \mathrm{e}^{(u_k L(u_k)/2)} \cdot \exp\left(\frac{L_c}{2} y_{ks} u_k\right) \cdot \chi_k(\grave{s}, s),\end{aligned} \tag{3.44}$$

where y_{ks} is the received version of the transmitted systematic bit $x_{k1} = u_k$ and:

$$\chi_k(\grave{s}, s) = \exp\left(\frac{L_c}{2} \sum_{l=2}^{n} y_{kl} x_{yl}\right). \tag{3.45}$$

Using Equation 3.44 and remembering that in the numerator we have $u_k = +1$ for all terms in the summation, whereas in the denominator we have $u_k = -1$, we can rewrite Equation 3.42 as:

$$\begin{aligned}L(u_k|\underline{y}) &= \ln\left(\frac{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=+1}} \alpha_{k-1}(\grave{s}) \cdot \gamma_k(\grave{s}, s) \cdot \beta_k(s)}{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=-1}} \alpha_{k-1}(\grave{s}) \gamma_k(\grave{s}, s) \cdot \beta_k(s)}\right) \\ &= \ln\left(\frac{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=+1}} \alpha_{k-1}(\grave{s}) \cdot \mathrm{e}^{+L(u_k)/2} \cdot \mathrm{e}^{+L_c y_{ks}/2} \cdot \chi_k(\grave{s}, s) \cdot \beta_k(s)}{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=-1}} \alpha_{k-1}(\grave{s}) \cdot \mathrm{e}^{-L(u_k)/2} \cdot \mathrm{e}^{-L_c y_{ks}/2} \cdot \chi_k(\grave{s}, s) \cdot \beta_k(s)}\right) \\ &= L(u_k) + L_c y_{ks} + \ln\left(\frac{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=+1}} \alpha_{k-1}(\grave{s}) \cdot \chi_k(\grave{s}, s) \cdot \beta_k(s)}{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=-1}} \alpha_{k-1}(\grave{s}) \cdot \chi_k(\grave{s}, s) \cdot \beta_k(s)}\right) \\ &= L(u_k) + L_c y_{ks} + L_e(u_k),\end{aligned} \tag{3.46}$$

where:

$$L_e(u_k) = \ln\left(\frac{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=+1}} \alpha_{k-1}(\grave{s}) \cdot \chi_k(\grave{s}, s) \cdot \beta_k(s)}{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=-1}} \alpha_{k-1}(\grave{s}) \cdot \chi_k(\grave{s}, s) \cdot \beta_k(s)}\right). \tag{3.47}$$

Thus we can see that the a-posteriori LLR $L(u_k|\underline{y})$ calculated with the aid of the MAP algorithm can be viewed as comprising three additive soft-metric terms:- $L(u_k)$, $L_c y_{ks}$ and $L_e(u_k)$. The first soft-metric term is the a-priori LLR $L(u_k)$, which accrues from $P(u_k)$ in the expression for the branch transition probability $\gamma_k(\grave{s}, s)$ in Equation 3.32. This probability should be generated by an independent source and is referred to as the a-priori probability of the kth information or systematic bit represented as $+1$ or -1,

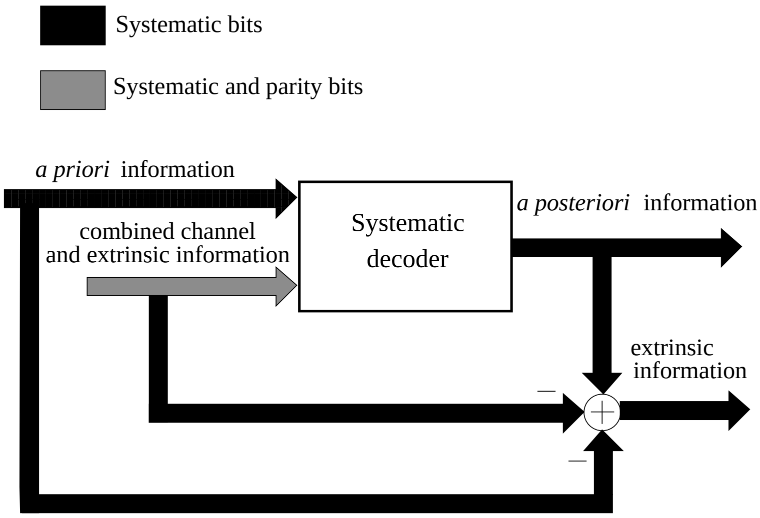


Figure 3.9: Schematic of a component decoder employed in a turbo decoder, showing the input information received and output information corresponding to the systematic and parity bits.

as illustrated in Figure 3.9. In most cases we will have no independent or a-priori knowledge of the likely value of the bit u_k at the decoder and hence the a-priori LLR $L(u_k)$ initially will be zero in the logarithmic domain, corresponding to an a-priori probability of $P(u_k) = 0.5$. However, in the case of an iterative turbo decoder, each component decoder is capable of providing the other decoder with an estimate of the a-priori LLR $L(u_k)$, as it will be described during our forthcoming discourse.

The second term $L_c y_{ks}$ in Equation 3.46 is the soft output of the channel representing the systematic bit u_k, which was directly transmitted across the channel and received as y_{ks}. In other words, this term corresponds to the systematic bits conveyed by the channel and to the extrinsic LLR values shown in Figure 3.9. When the channel SNR is high, the channel reliability value L_c of Equation 3.11 will be high and this systematic bit will have a large influence on the a-posteriori LLR $L(u_k|\underline{y})$. Conversely, when the channel SNR is low and hence L_c is low, the soft output of the channel for the received systematic bit y_{ks} will have less impact on the a-posteriori LLR delivered by the MAP algorithm.

The final term in Equation 3.46, $L_e(u_k)$, is derived using the constraints imposed by the code used, from the a-priori information sequence $L(u_n)$ and the received channel information sequence $\underline{y}$, *excluding* the received systematic bit y_{ks} and the a-priori information $L(u_k)$ for the bit u_k. Hence it is referred to as the *extrinsic* LLR for the bit u_k. Figure 3.9 and Equation 3.46 demonstrate that the extrinsic soft-information related to a specific bit and generated by a MAP decoder can be obtained by subtracting the a-priori soft-information value $L(u_k)$ and the received systematic soft output of the channel representing the systematic bit u_k - namely $L_c y_{ks}$ - from the soft output $L(u_k|\underline{y})$ of the decoder. This is the reason for having the subtraction paths shown in Figure 3.3. The corresponding formulae similar to Equation 3.46 can be derived for the other component decoder, which can be used in iterative turbo decoding. At this stage we underline again that the the a-priori soft-information value $L(u_k)$ and the received systematic soft output of the channel representing the systematic bit u_k - namely $L_c y_{ks}$ - **are subtracted from the soft output $L(u_k|\underline{y})$ of the decoder only for the systematic information bit u_k, but not for the parity bit**, as shown explicitly in Figure 3.9 with the aid of the black lines. By contrast, when using turbo equalisation, which will be the subject of Part III of the book, it will be shown in the context of Figure 9.3 that the extrinsic information has to be generated also for the parity bits.

Notice that the expression describing the branch transition probabilities $\gamma_k(\grave{s}, s)$ in Equation 3.40 uses the a-priori soft-information $L(u_k)$ and all n bits, including the systematic bit y_{ks} of the received codeword $\underline{y}_k$. Observe that these branch transition probabilities are used in the recursive calculations of $\alpha_k(s)$ and

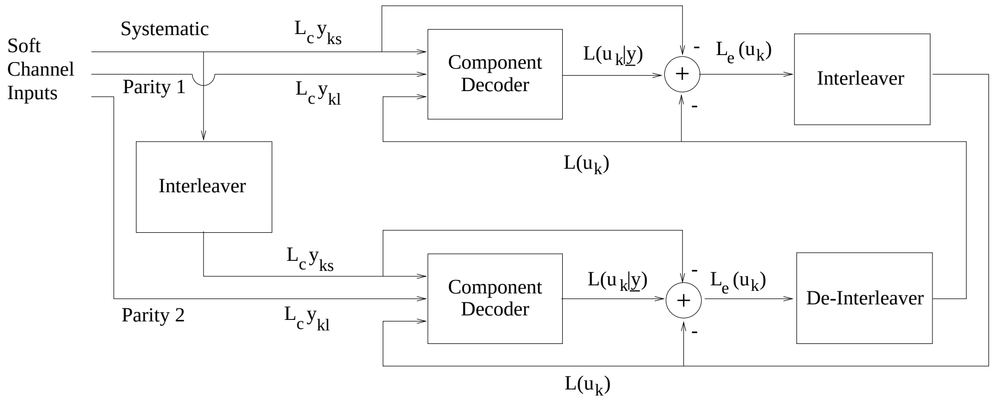


Figure 3.10: Turbo decoder schematic.

$\beta_k(s)$ in Equations 3.25 and 3.28. Hence these terms appear in Equation 3.47 describing $L_e(u_k)$ and therefore it might seem that the received systematic soft-bit y_{ks} and the a-priori soft-information $L(u_k)$ of the bit u_k appear indirectly in the extrinsic soft-output $L_e(u_k)$. However, careful examination of Equation 3.47 shows that for the bit u_k we use the values of $\alpha_{k-1}(\grave{s})$ and $\beta_k(s)$. From Equations 3.25 and 3.28 derived for the recursive calculation of these values we observe that the branch transition probabilities $\gamma_n(\grave{s}, s)$ for $1 \leq n \leq k-1$ and $k+1 \leq n \leq N$ will be used for calculating the $\alpha_{k-1}(\grave{s})$ and $\beta_k(s)$ values. Notice, however, that the specific branch transition probability $\gamma_k(\grave{s}, s)$, which characterises the transition associated with the bit u_k, is not used. MOre explicitly, $L_e(u_k)$ uses the values of the branch transition probabilities $\gamma_n(\grave{s}, s)$ for all the branches *except* for the kth branch. Therefore, although it does depend on all the other a-priori information terms $L(u_n)$ and on the received systematic bits, the term $L_e(u_k)$ is actually independent of the a-priori information $L(u_k)$ and the received systematic bit y_{ks}. Hence it can be justifiably referred to as the extrinsic LLR of the bit u_k.

We summarise below what is meant by the terms a-priori, extrinsic and a-posteriori information, which we use throughout this treatise.

a priori The a-priori information related to a bit is information known before decoding commences, from a source other than the received sequence or the code constraints. It is also sometimes referred to as intrinsic information for contrasting it with the extrinsic information to be described next.

extrinsic The extrinsic information related to a bit u_k is the information provided by a decoder based on the received sequence and on the a-priori information, but *excluding* the received systematic bit y_{ks} and the a-priori information $L(u_k)$ related to the bit u_k. Typically the component decoder provides this information using the constraints imposed on the transmitted sequence by the code used. It processes the received bits and the a-priori information surrounding the systematic bit u_k, and uses this information and the code constraints for providing information about the value of the bit u_k.

a posteriori The a-posteriori information related to a bit is the information that the decoder generates by taking into account *all* available sources of information concerning u_k. It is the a-posteriori LLR, ie $L(u_k|\underline{y})$, that the MAP algorithm generates as its output.

3.3.4.2 Iterative Turbo Decoding

We now describe how the iterative decoding of turbo codes is carried out. Figure 3.3 from Section 3.3.1, showing the structure of an iterative turbo decoder, is repeated for convenience here as Figure 3.10. Figure 3.10 also shows the symbols we have used for the various inputs to and outputs from the component decoders.

Consider initially the first component decoder in the first iteration. This decoder receives the channel sequence $L_c\underline{y}^{(1)}$ containing the received versions of the transmitted systematic bits, $L_c y_{ks}$, and the parity bits, $L_c y_{kl}$, from the first encoder. Usually, to obtain a half-rate code, half of these parity bits will have been punctured at the transmitter, and so the turbo decoder must insert zeros in the soft channel output $L_c y_{kl}$ for these punctured bits. The first component decoder can then process the soft channel inputs and produce its estimate $L_{11}(u_k|\underline{y})$ of the conditional LLRs of the data bits u_k, $k = 1, 2 \ldots N$. In this notation the subscript 11 in $L_{11}(u_k|\underline{y})$ indicates that this is the a-posteriori LLR in the first iteration from the first component decoder. Note that in this first iteration the first component decoder will have no a-priori information about the bits, and hence $L(u_k)$ in Equation 3.40 giving $\gamma_k(\grave{s}, s)$ will be zero, corresponding to an a-priori probability of 0.5.

Next the second component decoder comes into operation. It receives the channel sequence $L_c\underline{y}^{(2)}$ containing the *interleaved* version of the received systematic bits, and the parity bits from the second encoder. Again, the turbo decoder will have to insert zeros into this sequence, if the parity bits generated by the encoder are punctured before transmission. However, now, in addition to the received channel sequence $L_c\underline{y}^{(2)}$, the decoder can use the conditional LLR $L_{11}(u_k|\underline{y})$ provided by the first component decoder to generate a-priori LLRs $L(u_k)$ to be used by the second component decoder. Ideally these a-priori LLRs $L(u_k)$ would be completely independent from all the other information used by the second component decoder. As can be seen in Figure 3.10 in iterative turbo decoders the extrinsic information $L_e(u_k)$ from the other component decoder is used as the a-priori LLRs, after being interleaved to arrange the decoded data bits $\underline{u}$ in the same order as they were encoded by the second encoder. Again, according to Equation 3.46, the reason for the subtraction paths shown in Figure 3.10 is that the a-posteriori LLRs from one decoder have the systematic soft channel inputs $L_c y_{ks}$ and the a-priori LLRs $L(u_k)$ (if any were available) subtracted to yield the extrinsic LLRs $L_e(u_k)$ which are then used as a-priori LLRs for the other component decoder. The second component decoder thus uses the received channel sequence $L_c\underline{y}^{(2)}$ and the a-priori LLRs $L(u_k)$ (derived by interleaving the extrinsic LLRs $L_e(u_k)$ of the first component decoder) to produce its a-posteriori LLRs $L_{12}(u_k|\underline{y})$. This is then the end of the first iteration.

For the second iteration the first component encoder again processes its received channel sequence $L_c\underline{y}^{(1)}$, but now it also has a-priori LLRs $L(u_k)$ provided by the extrinsic portion $L_e(u_k)$ of the a-posteriori LLRs $L_{12}(u_k|\underline{y})$ calculated by the second component encoder, and hence it can produce an improved a-posteriori LLR $L_{21}(u_k|\underline{y})$. The second iteration then continues with the second component decoder using the improved a-posteriori LLRs $L_{21}(u_k|\underline{y})$ from the first encoder to derive, through Equation 3.46, improved a-priori LLRs $L(u_k)$ which it uses in conjunction with its received channel sequence $L_c\underline{y}^{(2)}$ to calculate $L_{22}(u_k|\underline{y})$.

This iterative process continues, and with each iteration on average the BER of the decoded bits will fall. However, as will be seen in Figure 3.21, the improvement in performance for each additional iteration carried out falls as the number of iterations increases. Hence for complexity reasons usually only about eight iterations are carried out, as no significant improvement in performance is obtained with a higher number of iterations. This is the arrangement we have used in most of our simulations, i.e. the decoder carries out a fixed number of iterations. However, it is possible to use a variable number of iterations up to a maximum, with some termination criterion used to decide when it is deemed that further iterations will produce marginal gain. This allows the average number of iterations, and so the average complexity of the decoder, to be dramatically reduced [61] with only a small degradation in performance. Suitable termination criteria have been found to be the so-called cross-entropy of the outputs from the two component decoders [61], and the variance of the a-posteriori LLRs $L(u_k|\underline{y})$ of a component decoder [150].

Figure 3.11 shows how the a-posteriori LLRs $L(u_k|\underline{y})$ output from the component decoders in an iterative decoder vary with the number of iterations used. The output from the second component decoder is shown after one, two, four and eight iterations. The input sequence of the encoder consisted entirely of logical 0s, and consequently the negative a-posteriori LLR $L(u_k|\underline{y})$ values correspond to a correct hard decision, while the positive values to an incorrect hard decision. The input sequence was coded using a turbo encoder with two constraint length 3 recursive convolutional codes, and a block interleaver with 31 rows and 31 columns. This turbo encoder is used in the majority of our investigations and its performance is characterised in Section 3.4. The encoded bits were transmitted over an AWGN channel at a channel SNR of -1 dB, and then decoded using an iterative turbo decoder using the MAP algorithm. It can be seen that as the number of iterations used increases, the number of positive a-posteriori LLR $L(u_k|\underline{y})$ values, and

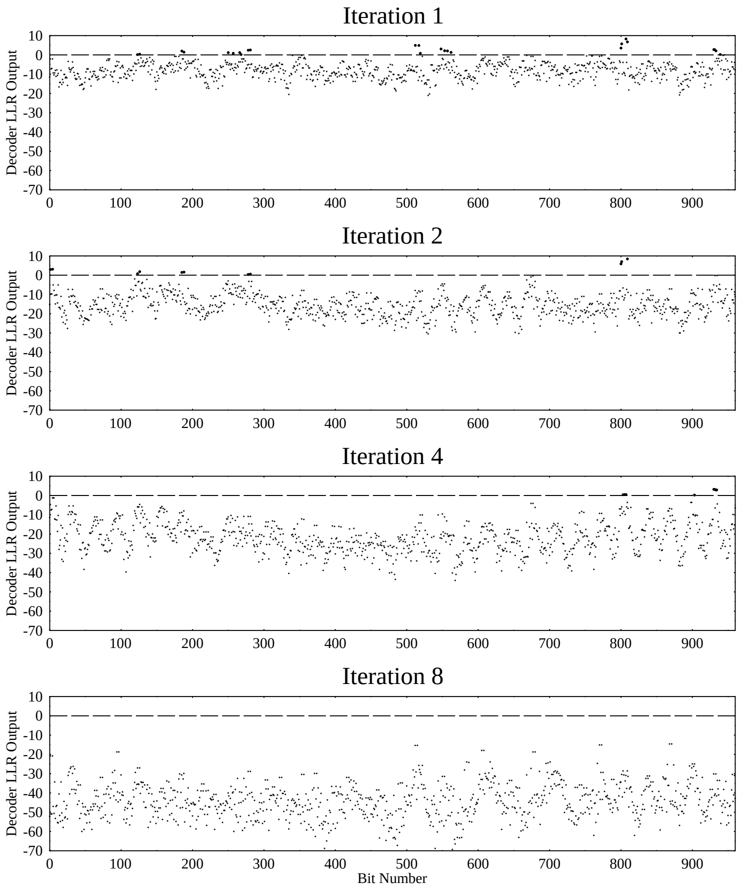


Figure 3.11: Soft outputs from the MAP decoder in an iterative turbo decoder for a transmitted stream of all -1.

hence the BER, decreases until after eight iterations there are no incorrectly decoded values. Furthermore, as the number of iterations increases, the decoders become more certain about the value of the bits and hence the magnitudes of the LLRs gradually become larger. The erroneous decisions in the figure appear in bursts, since deviating from the error-free trellis path typically inflicts several bit errors.

When the series of iterations halts, after either a fixed number of iterations or when a termination criterion is satisfied, the output from the turbo decoder is given by the de-interleaved a-posteriori LLRs of the second component decoder, $L_{i2}(u_k|\underline{y})$, where i is the number of iterations used. The sign of these a-posteriori LLRs gives the hard-decision output, i.e. whether the decoder believes that the transmitted data bit u_k was $+1$ or -1, and in some applications the magnitude of these LLRs, which gives the confidence the decoder has in its decision, may also be useful.

Ideally, for the iterative decoding of turbo codes, the a-priori information used by a component decoder should be completely independent from the channel outputs used by that decoder. However, in turbo decoders the extrinsic LLR $L_e(u_k)$ for the bit u_k, as explained above, uses all the available received parity bits and all the received systematic bits except the received value y_{ks} of the bit u_k. However, the same received systematic bits are also used by the other component decoder, which uses the interleaved or de-interleaved version of $L_e(u_k)$ as its a-priori LLRs. Hence the a-priori LLRs $L(u_k)$ are not truly independent from the channel outputs $\underline{y}$ used by the component decoders. However, owing to the fact that the component convolutional codes have a short memory, usually of only four bits or less, the extrinsic LLR $L_e(u_k)$ is only significantly affected by the received systematic bits relatively close to the bit u_k. When this extrinsic LLR $L_e(u_k)$ is used as the a-priori LLR $L(u_k)$ by the other component decoder, because of the interleaving used, the bit u_k and its neighbours will probably have been well separated. Hence the dependence of the a-priori LLRs $L(u_k)$ on the received systematic channel values $L_c y_{ks}$ which are also used by the other component decoder will have relatively little effect, and the iterative decoding provides good results.

Another justification for using the iterative arrangement described above is how well it has been found to work. In the limited experiments that have been carried out with optimal decoding of turbo codes [148, 149, 152] it has been found that optimal decoding performs only a fraction of a decibel (around 0.35–0.5 dB) better than iterative decoding with the MAP algorithm. Furthermore various turbo coding schemes have been found [63, 152] that approach the Shannonian limit, which gives the best performance theoretically available, to a similar fraction of a decibel. Therefore it seems that, for a variety of codes, the iterative decoding of turbo codes gives an almost optimal performance. Hence it is this iterative decoding structure, which is almost exclusively used with turbo codes, which we have used throughout our simulations.

Having described how the MAP algorithm can be used in the iterative decoding of turbo codes, we now proceed to describe other soft-in soft-out decoders, which are less complex and can be used instead of the MAP algorithm. We first describe two related algorithms, the Max-Log-MAP [50, 51, 153] and the Log-MAP [52], which are derived from the MAP algorithm, and then another, referred to as the Soft Output Viterbi Algorithm (SOVA) [53, 146], derived from the Viterbi algorithm.

3.3.5 Modifications of the MAP algorithm

3.3.5.1 Introduction

The MAP algorithm as described in Section 3.3.3 is much more complex than the Viterbi algorithm and with hard-decision outputs performs almost identically to it. Therefore for almost 20 years it was largely ignored. However, its application in turbo codes renewed interest in the algorithm, and it was realised that its complexity can be dramatically reduced without affecting its performance. The roots of employing the Max-Log-MAP algorithm based on the Jacobian logarithmic approximation to be outlined in Equation 3.56 go back to the pre-1993 era, preceding the publication of the turbo coding principles by Berrou *et al.* [12]. More specifically, to the authors' knowledge the Jacobian logarithmic approximation has been used in this context for the first time in 1989 in the master thesis of Erfanian at the University of Toronto [3]. and in the open literature in references [50, 51, 153]. The Max-Log-MAP technique simplified the MAP algorithm by transferring the recursions into the logarithmic domain and by invoking an approximation for the sake of dramatically reducing the associated implementational complexity. Because of this approximation the performance of the Max-Log-MAP algorithms is sub-optimal. However, Robertson *et al.* [52] in 1995 proposed the Log-MAP algorithm, which corrected the approximation used in the Max-Log-MAP algorithm and hence attained a performance virtually identical to that of the MAP algorithm at a fraction of its complexity. These two algorithms are described in more detail in this section.

3.3.5.2 Mathematical Description of the Max-Log-MAP Algorithm

The MAP algorithm calculates the a-posteriori LLRs $L(u_k|\underline{y})$ using Equation 3.42. To do this it requires the following values:

1) The $\alpha_{k-1}(\grave{s})$ values, which are calculated in a forward recursive manner using Equation 3.25,
2) the $\beta_k(s)$ values, which are calculated in a backward recursion using Equation 3.28, and
3) the branch transition probabilities $\gamma_k(\grave{s}, s)$, which are calculated using Equation 3.40.

The Max-Log-MAP algorithm simplifies this by transferring these equations into the log arithmetic domain and then using the approximation:

$$\ln\left(\sum_i \mathrm{e}^{x_i}\right) \approx \max_i(x_i), \tag{3.48}$$

[3]Private communication by Erfanian and Pasupathy.

where $\max_i(x_i)$ means the maximum value of x_i. Then, with $A_k(s)$, $B_k(s)$ and $\Gamma_k(\grave{s}, s)$ defined as follows:

$$A_k(s) \triangleq \ln(\alpha_k(s)), \tag{3.49}$$

$$B_k(s) \triangleq \ln(\beta_k(s)), \tag{3.50}$$

and:

$$\Gamma_k(\grave{s}, s) \triangleq \ln(\gamma_k(\grave{s}, s)), \tag{3.51}$$

we can rewrite Equation 3.25 as:

$$\begin{aligned} A_k(s) &\triangleq \ln(\alpha_k(s)) \\ &= \ln\left(\sum_{\text{all } \grave{s}} \alpha_{k-1}(\grave{s})\gamma_k(\grave{s}, s)\right) \\ &= \ln\left(\sum_{\text{all } \grave{s}} \exp\left[A_{k-1}(\grave{s}) + \Gamma_k(\grave{s}, s)\right]\right) \\ &\approx \max_{\grave{s}}\left(A_{k-1}(\grave{s}) + \Gamma_k(\grave{s}, s)\right). \end{aligned} \tag{3.52}$$

Equation 3.52 implies that for each path in Figure 3.6 from the previous stage in the trellis to the state $S_k = s$ at the present stage, the algorithm adds a branch metric term $\Gamma_k(\grave{s}, s)$ to the previous value $A_{k-1}(\grave{s})$ to find a new value $\tilde{A}_k(s)$ for that path. The new value of $A_k(s)$ according to Equation 3.52 is then the maximum of the $\tilde{A}_k(s)$ values of the various paths reaching the state $S_k = s$. This can be thought of as selecting one path as the 'survivor' and discarding any other paths reaching the state.

The value of $A_k(s)$ should give the natural logarithm of the probability that the trellis is in state $S_k = s$ at stage k, given that the received channel sequence up to this point has been $\underline{y}_{j \leq k}$. However, because of the approximation of Equation 3.48 used to derive Equation 3.52, only the maximum likelihood path through the state $S_k = s$ is considered when calculating this probability. Thus the value of A_k in the Max-Log-MAP algorithm actually gives the probability of the most likely path through the trellis to the state $S_k = s$, rather than the probability of *any* path through the trellis to state $S_k = s$. This approximation is one of the reasons for the sub-optimal performance of the Max-Log-MAP algorithm compared to the MAP algorithm.

We see from Equation 3.52 that in the Max-Log-MAP algorithm the forward recursion used to calculate $A_k(s)$ is exactly the same as the forward recursion in the Viterbi algorithm — for each pair of merging paths the survivor is found using two additions and one comparison. Notice that for binary trellises the summation, and maximisation, over all previous states $S_{k-1} = \grave{s}$ in Equation 3.52 will in fact be over only two states, because there will be only two previous states $S_{k-1} = \grave{s}$ with paths to the present state $S_k = s$. For all other values of $\grave{s}$ we will have $\gamma_k(\grave{s}, s) = 0$.

Similarly to Equation 3.52 for the forward recursion used to calculate the $A_k(s)$, we can rewrite Equation 3.28 as:

$$\begin{aligned} B_{k-1}(\grave{s}) &\triangleq \ln(\beta_{k-1}(\grave{s})) \\ &= \ln\left(\sum_{\text{all } s} \beta_k(s)\gamma_k(\grave{s}, s)\right) \\ &= \ln\left(\sum_{\text{all } s} \exp\left[B_k(s) + \Gamma_k(\grave{s}, s)\right]\right) \\ &\approx \max_{s}\left(B_k(s) + \Gamma_k(\grave{s}, s)\right), \end{aligned} \tag{3.53}$$

and obtain the backward recursion used to calculate the $B_{k-1}(\grave{s})$ values. Again this is equivalent to the recursion used in the Viterbi algorithm — the value of $B_{k-1}(\grave{s})$ is found by adding, for every state $S_k = s$ having a path from $S_{k-1} = \grave{s}$ (two in a binary trellis), a branch metric $\Gamma_k(\grave{s}, s)$ to the value of $B_k(s)$ and selecting which path gives the highest $B_{k-1}(\grave{s})$ value.

Using Equation 3.40, the branch metrics $\Gamma_k(\grave{s}, s)$ in the recursive formulae of Equations 3.52 and 3.53 derived for $A_k(s)$ and $B_{k-1}(\grave{s})$, respectively, can be written as:

$$\begin{aligned}
\Gamma_k(\grave{s}, s) &\triangleq \ln\left(\gamma_k(\grave{s}, s)\right) \\
&= \ln\left(C \cdot \mathrm{e}^{(u_k L(u_k)/2)} \exp\left[\frac{E_b}{2\sigma^2} 2a \sum_{l=1}^{n} y_{kl} x_{kl}\right]\right) \\
&= \ln\left(C \cdot \mathrm{e}^{(u_k L(u_k)/2)} \exp\left[\frac{L_c}{2} \sum_{l=1}^{n} y_{kl} x_{kl}\right]\right) \\
&= \hat{C} + \frac{1}{2} u_k L(u_k) + \frac{L_c}{2} \sum_{l=1}^{n} y_{kl} x_{kl},
\end{aligned} \tag{3.54}$$

where $\hat{C} = \ln C$ does not depend on u_k or on the transmitted codeword $\underline{x}_k$ and so can be considered a constant and omitted. Hence the branch metric is equivalent to that used in the Viterbi algorithm, with the addition of the a-priori LLR term $u_k L(u_k)$. Furthermore the correlation term $\sum_{l=1}^{n} y_{kl} x_{kl}$ is weighted by the channel reliability value L_c of Equation 3.11.

Finally, from Equation 3.42, we can write for the a-posteriori LLRs $L(u_k|\underline{y})$ which the Max-Log-MAP algorithm calculates:

$$\begin{aligned}
L(u_k|\underline{y}) &= \ln\left(\frac{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=+1}} \alpha_{k-1}(\grave{s}) \cdot \gamma_k(\grave{s}, s) \cdot \beta_k(s)}{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=-1}} \alpha_{k-1}(\grave{s}) \cdot \gamma_k(\grave{s}, s) \cdot \beta_k(s)}\right) \\
&= \ln\left(\frac{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=+1}} \exp\left(A_{k-1}(\grave{s}) + \Gamma_k(\grave{s}, s) + B_k(s)\right)}{\sum\limits_{\substack{(\grave{s},s)\Rightarrow \\ u_k=-1}} \exp\left(A_{k-1}(\grave{s}) + \Gamma_k(\grave{s}, s) + B_k(s)\right)}\right) \\
&\approx \max_{\substack{(\grave{s},s)\Rightarrow \\ u_k=+1}} \left(A_{k-1}(\grave{s}) + \Gamma_k(\grave{s}, s) + B_k(s)\right) \\
&- \max_{\substack{(\grave{s},s)\Rightarrow \\ u_k=-1}} \left(A_{k-1}(\grave{s}) + \Gamma_k(\grave{s}, s) + B_k(s)\right).
\end{aligned} \tag{3.55}$$

This means that in the Max-Log-MAP algorithm for each bit u_k the a-posteriori LLR $L(u_k|\underline{y})$ is calculated by considering every transition from the trellis stage S_{k-1} to the stage S_k. These transitions are grouped into those that might have occurred if $u_k = +1$, and those that might have occurred if $u_k = -1$. For both of these groups the transition giving the maximum value of $A_{k-1}(\grave{s}) + \Gamma_(\grave{s}, s) + B_k(s)$ is found, and the a-posteriori LLR is calculated based on only these two 'best' transitions. For a binary trellis there will be $2 \cdot 2^{K-1}$ transitions at each stage of the trellis, where K is the constraint length of the convolutional code. Therefore there will be 2^{K-1} transitions to consider in each of the maximisations in Equation 3.55.

The Max-Log-MAP algorithm can be summarised as follows. Forward and backward recursions, both similar to the forward recursion used in the Viterbi algorithm, are used to calculate $A_k(s)$ using Equation 3.52 and $B_k(s)$ using Equation 3.53. The branch metric $\Gamma_k(\grave{s}, s)$ used is given by Equation 3.54, where the constant term $\hat{C}$ can be omitted. Once both the forward and backward recursions have been carried out, the a-posteriori LLRs can be calculated using Equation 3.55. Thus the complexity of the Max-Log-MAP algorithm is not hugely higher than that of the Viterbi algorithm — instead of one recursion two are carried out, the branch metric of Equation 3.54 has the additional a-priori term $u_k L(u_k)$ term added to it, and for each bit Equation 3.55 must be used to give the a-posteriori LLRs. This calculation of $L(u_k|\underline{y})$ from the $A_{k-1}(\grave{s})$, $B_k(s)$ and $\Gamma_k(\grave{s}, s)$ values requires, for every bit 2 additions for each of the $2 \cdot 2^{K-1}$ transitions at each stage of the trellis, two maximisations and one subtraction. Viterbi states [154] that the complexity of the Log-MAP-Max algorithm is no greater than three times that of a Viterbi decoder. Unfortunately the storage requirements are much greater because of the need to store both the forward and backward recursively calculated metrics $A_k(s)$ and $B_k(s)$ before the $L(u_k|\underline{y})$ values can be calculated. However, Viterbi also states [154, 155] that it can be shown that by increasing the computational load slightly the associated memory requirements can be dramatically reduced.

3.3.5.3 Correcting the Approximation — the Log-MAP Algorithm

The Max-Log-MAP algorithm gives a slight degradation in performance compared to the MAP algorithm owing to the approximation of Equation 3.48. When used for the iterative decoding of turbo codes, this degradation was found by Robertson *et al.* [52] to result in a drop in performance of about 0.35 dB. However, the approximation of Equation 3.48 can be made exact by using the Jacobian logarithm:

$$\begin{aligned} \ln\left(\mathrm{e}^{x_1} + \mathrm{e}^{x_2}\right) &= \max(x_1, x_2) + \ln\left(1 + \mathrm{e}^{-|x_1 - x_2|}\right) \\ &= \max(x_1, x_2) + f_c\left(|x_1 - x_2|\right) \\ &= g(x_1, x_2), \end{aligned} \tag{3.56}$$

where $f_c(x)$ can be thought of as a correction term. This is then the basis of the Log-MAP algorithm proposed by Robertson *et al.* [52]. Similarly to the Max-Log-MAP algorithm, values for $A_k(s) \triangleq \ln(\alpha_k(s))$ and $B_k(s) \triangleq \ln(\beta_k(s))$ are calculated using a forward and a backward recursion. However, the maximisation in Equations 3.52 and 3.53 is complemented by the correction term in Equation 3.56. This means that the exact rather than approximate values of $A_k(s)$ and $B_k(s)$ are calculated. In binary trellises, as explained earlier, the maximisation will be over only two terms. Therefore we can correct the approximations in Equations 3.52 and 3.53 by merely adding the term $f_c(\delta)$, where δ is the magnitude of the difference between the metrics of the two merging paths. Similarly, the approximation in Equation 3.55 giving the a-posteriori LLRs $L(u_k|\underline{y})$ can be eliminated using the Jacobian logarithm. However, as explained earlier, there will be 2^{K-1} transitions to consider in each of the maximisations of Equation 3.55. Thus we must generalise Equation 3.48 in order to cope with more than two x_i terms. This is done by nesting the $g(x_1, x_2)$ operations as follows:

$$\ln\left(\sum_{i=1}^{I} e^{x_i}\right) = g(x_I, g(x_{I-1}, \cdots, g(x_3, g(x_2, x_1))) \cdots). \tag{3.57}$$

The correction term $f_c(\delta)$ need not be computed for every value of δ, but instead can be stored in a look-up table. Robertson *et al.* [52] found that such a look-up table need contain only eight values for δ, ranging between 0 and 5. This means that the Log-MAP algorithm is only slightly more complex than the Max-Log-MAP algorithm, but it gives exactly the same performance as the MAP algorithm. Therefore it is a very attractive algorithm to use in the component decoders of an iterative turbo decoder.

Having described two techniques based on the MAP algorithm but with reduced complexity, we now describe an alternative soft-in soft-out decoder based on the Viterbi algorithm.

3.3.6 The Soft-output Viterbi Algorithm

3.3.6.1 Mathematical Description of the Soft-output Viterbi Algorithm

In this section we describe a variation of the Viterbi algorithm, referred to as the Soft-Output Viterbi Algorithm (SOVA) [53, 146]. This algorithm has two modifications over the classical Viterbi algorithm which allow it to be used as a component decoder for turbo codes. First the path metrics used are modified to take account of a-priori information when selecting the maximum likelihood path through the trellis. Second the algorithm is modified so that it provides a soft output in the form of the a-posteriori LLR $L(u_k|\underline{y})$ for each decoded bit.

The first modification is easily accomplished. Consider the state sequence $\underline{s}_k^s$ which gives the states along the surviving path at state $S_k = s$ at stage k in the trellis. The probability that this is the correct path through the trellis is given by:

$$p(\underline{s}_k^s | \underline{y}_{j \leq k}) = \frac{p(\underline{s}_k^s \wedge \underline{y}_{j \leq k})}{p(\underline{y}_{j \leq k})}. \tag{3.58}$$

As the probability of the received sequence $\underline{y}_{j \leq k}$ for transitions up to and including the kth transition is constant for all paths $\underline{s}_k$ through the trellis to stage k, the probability that the path $\underline{s}_k^s$ is the correct one is proportional to $p(\underline{s}_k^s \wedge \underline{y}_{j \leq k})$. Therefore our metric should be defined so that maximising the metric will maximise $p(\underline{s}_k^s \wedge \underline{y}_{j \leq k})$. The metric should also be easily computable in a recursive manner as we go from the $(k-1)$th stage in the trellis to the kth stage. If the path $\underline{s}_k^s$ at the kth stage has the path $\underline{s}_{k-1}^{\grave{s}}$ for its first $k-1$ transitions then, assuming a memoryless channel, we will have:

$$p(\underline{s}_k^s \wedge \underline{y}_{j \leq k}) = p(\underline{s}_{k-1}^{\grave{s}} \wedge \underline{y}_{j \leq k-1}) \cdot p(S_k = s \wedge \underline{y}_k | S_{k-1} = \grave{s}). \tag{3.59}$$

A suitable metric for the path $\underline{s}_k^s$ is therefore $M(\underline{s}_k^s)$, where

$$\begin{aligned} M(\underline{s}_k^s) &\triangleq \ln\left(p(\underline{s}_k^s \wedge \underline{y}_{j \leq k})\right) \\ &= M(\underline{s}_{k-1}^{\grave{s}}) + \ln\left(\mathrm{p}(\mathrm{S_k = s} \wedge \underline{\mathrm{y}}_\mathrm{k} | \mathrm{S_{k-1}} = \grave{\mathrm{s}})\right). \end{aligned} \tag{3.60}$$

Using Equation 3.23 we then have:

$$M(\underline{s}_k^s) = M(\underline{s}_{k-1}^{\grave{s}}) + \ln\left(\gamma_\mathrm{k}(\grave{\mathrm{s}}, \mathrm{s})\right), \tag{3.61}$$

where $\gamma_k(\grave{s}, s)$ is the branch transition probability for the path from $S_{k-1} = \grave{s}$ to $S_k = s$. From Equation 3.54 we can write:

$$\ln\left(\gamma_k(\grave{s}, s)\right) \triangleq \Gamma_k(\grave{s}, s) = \hat{C} + \frac{1}{2} u_k L(u_k) + \frac{L_c}{2} \sum_{l=1}^{n} y_{kl} x_{kl}, \tag{3.62}$$

and as the term $\hat{C}$ is constant, it can be omitted and we can rewrite Equation 3.61 as:

$$M(\underline{s}_k^s) = M(\underline{s}_{k-1}^{\grave{s}}) + \frac{1}{2} u_k L(u_k) + \frac{L_c}{2} \sum_{l=1}^{n} y_{kl} x_{kl}. \tag{3.63}$$

Hence our metric in the SOVA is updated as in the Viterbi algorithm, with the additional $u_k L(u_k)$ term included so that the a-priori information available is taken into account. Notice that this is equivalent to the forward recursion in Equation 3.52 used to calculate $A_k(s)$ in the Max-Log-MAP algorithm.

The possibility of modifying the metric used in the Viterbi algorithm to include a-priori information was mentioned by Forney [9] in his 1973 paper, although he proposed no application for such a modification. However, the requirement to use a-priori information in the soft-in soft-out component decoders of turbo decoders has provided an obvious application.

Let us now discuss the second modification of the algorithm required, i.e. to give soft outputs. In a binary trellis there will be two paths reaching state $S_k = s$ at stage k in the trellis. The modified Viterbi algorithm, which takes account of the a-priori information $u_k L(u_k)$, calculates the metric from Equation 3.63 for both merging paths, and discards the path with the lower metric. If the two paths $\underline{s}_k^s$ and $\underline{\hat{s}}_k^s$ reaching state $S_k = s$ have metrics $M(\underline{s}_k^s)$ and $M(\underline{\hat{s}}_k^s)$, and the path $\underline{s}_k^s$ is selected as the survivor because its metric is higher, then we can define the metric difference Δ_k^s as:

$$\Delta_k^s = M(\underline{s}_k^s) - M(\underline{\hat{s}}_k^s) \geq 0. \tag{3.64}$$

The probability that we have made the correct decision when we selected path $\underline{s}_k^s$ as the survivor and discarded path $\underline{\hat{s}}_k^s$ is then:

$$P(\text{correct decision at } S_k = s) = \frac{P(\underline{s}_k^s)}{P(\underline{s}_k^s) + P(\underline{\hat{s}}_k^s)}. \tag{3.65}$$

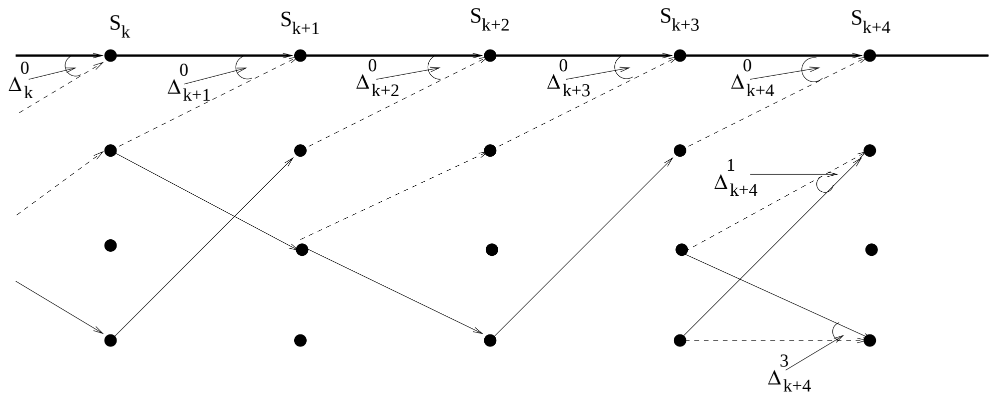


Figure 3.12: Simplified section of the trellis for our $K = 3$ RSC code with SOVA decoding.

Upon taking into account our metric definition in Equation 3.60 we have:

$$\begin{aligned} P(\text{correct decision at } S_k = s) &= \frac{e^{M(\underline{s}_k^s)}}{e^{M(\underline{s}_k^s)} + e^{M(\hat{\underline{s}}_k^s)}} \\ &= \frac{e^{\Delta_k^s}}{1 + e^{\Delta_k^s}}, \end{aligned} \tag{3.66}$$

and the LLR that this is the correct decision is given by:

$$\begin{aligned} L(\text{correct decision at } S_k = s) &= \ln\left(\frac{P(\text{correct decision at } S_k = s)}{1 - P(\text{correct decision at } S_k = s)}\right) \\ &= \Delta_k^s. \end{aligned} \tag{3.67}$$

Figure 3.12 shows a simplified section of the trellis for our $K = 3$ RSC code, with the metric differences Δ_k^s marked at various points in the trellis.

When we reach the end of the trellis and have identified the Maximum Likelihood (ML) path through the trellis, we need to find the LLRs giving the reliability of the bit decisions along the ML path. Observations of the Viterbi algorithm have shown that all the surviving paths at a stage l in the trellis will normally have come from the same path at some point before l in the trellis. This point is taken to be at most δ transitions before l, where usually δ is set to be five times the constraint length of the convolutional code. Therefore the value of the bit u_k associated with the transition from state $S_{k-1} = \grave{s}$ to state $S_k = s$ on the ML path may have been different if, instead of the ML path, the Viterbi algorithm had selected one of the paths which merged with the ML path up to δ transitions later, i.e. up to the trellis stage $k + \sigma$. By the arguments above, if the algorithm had selected any of the paths which merged with the ML path after this point the value of u_k would not be affected, because such paths will have diverged from the ML path after the transition from $S_{k-1} = \grave{s}$ to $S_k = s$. Thus, when calculating the LLR of the bit u_k, the SOVA must take account of the probability that the paths merging with the ML path from stage k to stage $k + \delta$ in the trellis were incorrectly discarded. This is done by considering the values of the metric difference $\Delta_i^{s_i}$ for all states s_i along the ML path from trellis stage $i = k$ to $i = k + \delta$. It is shown by Hagenauer in [54] that this LLR can be approximated by:

$$L(u_k|\underline{y}) \approx u_k \min_{\substack{i=k\cdots k+\delta \\ u_k \neq u_k^i}} \Delta_i^{s_i}, \tag{3.68}$$

where u_k is the value of the bit given by the ML path, and u_k^i is the value of this bit for the path which merged with the ML path and was discarded at trellis stage i. Thus the minimisation in Equation 3.68 is carried out only for those paths merging with the ML path which would have given a different value for the bit u_k if they had been selected as the survivor. The paths which merge with the ML path, but would have given the same value for u_k as the ML path, obviously do not affect the reliability of the decision of u_k.

For clarification of these operations refer again to Figure 3.12 showing a simplified section of the trellis for our $K = 3$ RSC code. In this figure, as before, continous lines represent transitions taken when the input bit is a -1, and broken lines represent transitions taken when the input bit is a +1. We assume that the all-zero path is identified as the ML path, and this path is shown as a bold line. Also shown are the paths which merge with this ML path. It can be seen from the figure that the ML path gives a value of -1 for u_k, but the paths merging with the ML path at trellis stages S_k, S_{k+1}, S_{k+3} and S_{k+4} all give a value of +1 for the bit u_k. Hence, if we assume for simplicity that $\sigma = 4$, from Equation 3.68 the LLR $L(u_k|\underline{y})$ will be given by -1 multiplied by the minimum of the metric differences Δ_k^0, Δ_{k+1}^0, Δ_{k+3}^0 and Δ_{k+4}^0.

3.3.6.2 Implementation of the SOVA

The SOVA can be implemented as follows. For each state at each stage in the trellis the metric $M(\underline{s}_k^s)$ is calculated for both of the two paths merging into the state using Equation 3.63. The path with the highest metric is selected as the survivor, and for this state at this stage in the trellis a pointer to the previous state along the surviving path is stored, just as in the classical Viterbi algorithm. However, in order to allow the reliability of the decoded bits to be calculated, the information used in Equation 3.68 to give $L(u_k|\underline{y})$ is also stored. Thus the difference Δ_k^s between the metrics of the surviving and the discarded paths is stored, together with a binary vector containing $\delta + 1$ bits, which indicate whether or not the discarded path would have given the same series of bits u_l for $l = k$ back to $l = k - \delta$ as the surviving path does. This series of bits is called the update sequence in [54], and as noted by Hagenauer it is given by the result of a modulo-2 addition (i.e. an exclusive-or operation) between the previous $\delta + 1$ decoded bits along the surviving and discarded paths. When the SOVA has identified the ML path, the stored update sequences and metric differences along this path are used in Equation 3.68 to calculate the values of $L(u_k|\underline{y})$.

The SOVA described in this section is the least complex of all the soft-in soft-out decoders discussed in this chapter. In [52] it is shown by Robertson *et al.* that the SOVA is about half as complex as the Max-Log-MAP algorithm. However, the SOVA is also the least accurate of the algorithms we have described in this chapter and, when used in an iterative turbo decoder, performs about 0.6 dB worse [52] than a decoder using the MAP algorithm. Figure 3.13 compares the LLRs output from the component decoders in an iterative turbo decoder using both the MAP and the SOVA algorithms. The same encoder, all -1 input sequence, and channel SNR as described for Figure 3.11 were used. It can be seen that the outputs of the SOVA are significantly more noisy than those from the MAP algorithm. For the second decoder at the eighth iteration the MAP algorithm gives LLRs which are all negative, and hence gives no bit errors. However, it can be seen from Figure 3.11 that, even after eight iterations, the SOVA still gives some positive LLRs, and hence will make several bit errors.

Let us now augment our understanding of iterative turbo decoding by considering a specific example.

3.3.7 Turbo Decoding Example

In this section we discuss an example of turbo decoding using the SOVA detailed in Section 3.3.6. This example serves to illustrate the details of the SOVA and the iterative decoding of turbo codes discussed in Section 3.3.4.

We consider a simple half-rate turbo code using the $K = 3$ RSC code, with generator polynomials expressed in octal form as 7 and 5, shown in Figure 3.2. Two such codes are combined, as shown in Figure 3.1, with a 3×3 block interleaver to give a simple turbo code. The parity bits from both the component codes are punctured, so that alternate parity bits from the first and the second component encoder are transmitted. Thus the first, third, fifth, seventh and ninth parity bits from the first component encoder are transmitted, and the second, fourth, sixth and eighth parity bits from the second component encoder are transmitted. The first component encoder is terminated using two bits chosen to take this encoder back to the all-zero state. The transmitted sequence will therefore contain nine systematic and nine parity bits. Of the systematic bits seven will be the input bits, and two will be the bits chosen to terminate the first trellis. Of the nine parity bits five will come from the first encoder, and four from the second encoder.

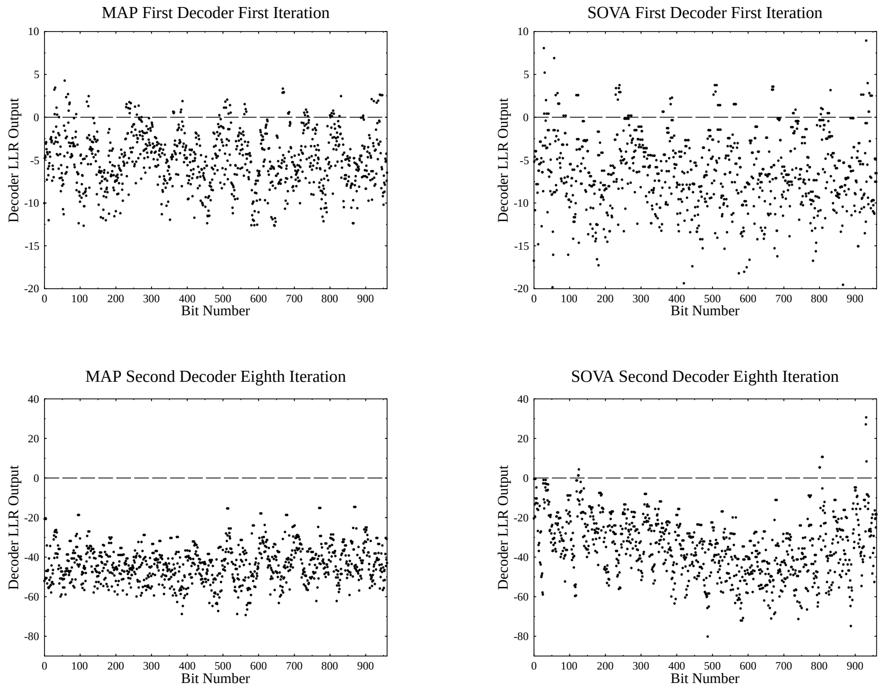


Figure 3.13: Soft outputs from the SOVA compared to the MAP algorithm for a transmitted stream of all -1

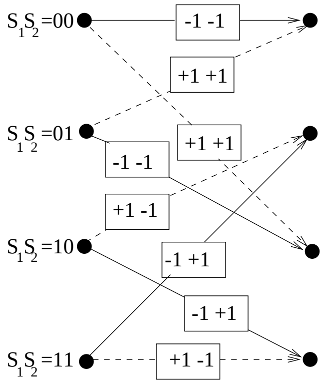


Figure 3.14: State transition diagram for our (2,1,3) RSC component codes.

Input Bit	Systematic Bit	Parity Bits Coder 1	Coder 2	Transmitted Sequence	Received Sequence
-1	-1	-1	–	$-1,-1$	$-2.1, -0.1$
-1	-1	–	-1	$-1, -1$	$-1.4, -1.4$
-1	-1	-1	–	$-1, -1$	$-1.7, -0.5$
-1	-1	–	-1	$-1, -1$	$+0.9, +0.5$
-1	-1	-1	–	$-1, -1$	$+1.2, -1.7$
-1	-1	–	-1	$-1, -1$	$-1.1, -1.1$
-1	-1	-1	–	$-1, -1$	$-0.7, -0.8$
-	-1	–	-1	$-1, -1$	$-2.4, -1.9$
-	-1	-1	–	$-1, -1$	$-1.6, -0.9$

Table 3.1: Input and transmitted bits for turbo decoding example.

The state transition diagram for the component RSC codes is shown in Figure 3.14. As in all our diagrams in this section, a continous line denotes a transition resulting from a -1 input bit, and a broken lines represents an input bit of +1. The figures within the boxes along the transition lines give the output bits associated with that transition:- the first bit is the systematic bit, which is the same as the input bit, and the second is the parity bit.

For the sake of simplicity we assume that an all -1 input sequence is used. Thus there will be seven input bits which are -1, and the encoder trellis will remain in the $S_1S_2 = 00$ state. The two bits necessary to terminate the trellis will be -1 in this case and, as can be seen from Figure 3.14, the resulting parity bits will also be -1. Thus all 18 of the transmitted bits will be -1 for an all -1 input sequence. Assuming that BPSK modulation is used with the transmitted symbols being -1 or +1, the transmitted sequence will be a series of $18 - 1$s. The received channel output sequence for our example, together with the input and transmitted bits detailed above, is shown in Table 3.1. Notice that approximately half the parity bits from each component encoder are punctured — this is represented by a dash in Table 3.1. Also note that the received channel sequence values shown in Table 3.1 are the matched filter outputs, which were denoted by y_{kl} in previous sections. If hard-decision demodulation were used then negative values would be decoded as -1s, and positive values as +1s. It can be seen that from the 18 coded bits which were transmitted, all of which were -1, three would be decoded as +1 if hard-decision demodulation were used.

In order to illustrate the difference between iterative turbo decoding and the decoding of convolutional codes, we initially consider how the received sequence shown in Table 3.1 would be decoded by a convolutional decoder using the Viterbi algorithm. Imagine the half-rate $K = 3$ RSC code detailed above used as an ordinary convolutional code to encode an input sequence of seven -1s. If trellis termination was used then two -1s would be employed to terminate the trellis, and the transmitted sequence would consist of 18 -1s, just as for our turbo coding example. If the received sequence was as shown in Table 3.1, then the Viterbi algorithm decoding this sequence would have the trellis diagram shown in Figure 3.15. The metrics shown in this figure are given by the cross-correlation of the received and expected channel sequences for a given path, and the Viterbi algorithm maximises this metric to find the ML path, which is shown by the bold line in Figure 3.15. Notice that at each state in the trellis where two paths merge, the path with the lower metric is discarded and its metric is shown crossed out in the figure. As can be seen from Figure 3.15, the Viterbi algorithm makes an incorrect decision at stage $k = 6$ in the trellis and selects a path other than the all-zero path as the survivor. This results in three of the seven bits being decoded incorrectly as +1s.

Having seen how Viterbi decoding of a RSC code would fail and produce three errors given our received sequence, we now proceed to detail the operation of an iterative turbo decoder for the same channel sequence. Consider first the operation of the first component decoder in the first iteration. The component decoder uses the SOVA to decide upon not only the most likely input bits, but also the LLRs of these bits, as described in Section 3.3.6. We will describe here how the SOVA calculates these LLRs for the channel values given in Table 3.1.

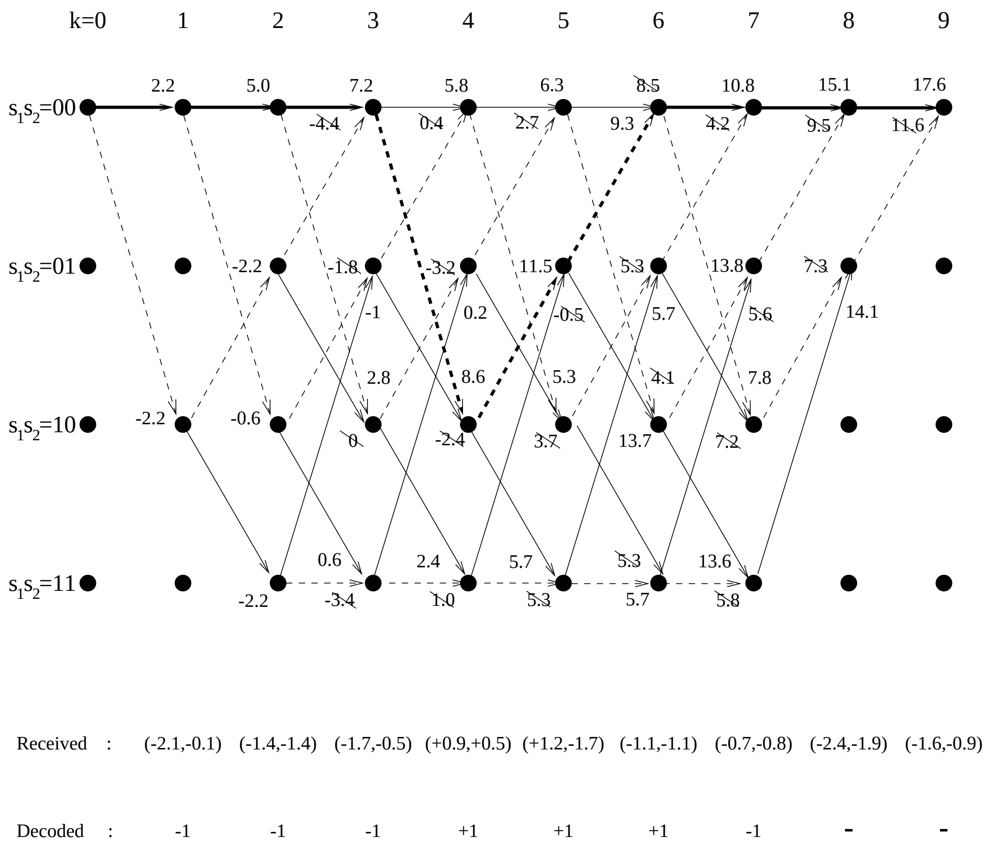


Figure 3.15: Trellis diagram for the Viterbi decoding of the received sequence shown in Table 3.1.

The metric for the SOVA is given by Equation 3.63, which is repeated here for convenience:

$$M(\underline{s}_k^s) = M(\underline{s}_{k-1}^{\grave{s}}) + \frac{1}{2}u_k L(u_k) + \frac{L_c}{2}\sum_{l=1}^{Q} y_{kl}x_{kl}. \tag{3.69}$$

Here $M(\underline{s}_{k-1}^{\grave{s}})$ is the metric for the surviving path through the state $S_{k-1} = \grave{s}$ at stage $k-1$ in the trellis, u_k and x_{kl} are the input bit and the transmitted channel sequence associated with a given transition, y_{kl} is the received channel sequence for that transition and L_c is the channel reliability value, as defined in Equation 3.11. As initially we are considering the operation of the first decoder in the first iteration there is no a-priori information and hence we have $L(u_k) = 0$ for all k, which corresponds to an a-priori probability of 0.5. The received sequence given in Table 3.1 was derived from the transmitted channel sequence (which has $E_b = 1$) by adding AWGN with variance $\sigma = 1$. Hence, as the fading amplitude is $a = 1$, from Equation 3.11 we have for the channel reliability measure $L_c = 2$.

Figure 3.16 shows the trellis for this first component decoder in the first iteration. Owing to the puncturing of the parity bits used at the encoder, the second, fourth, sixth and eighth parity bits have been received as zeros. The a-priori and channel values shown in Figure 3.16 are given as $L(u_k)/2$ and $L_c y_{kl}/2$ so that the metric values, given by Equation 3.69, can be calculated by simple addition and subtraction of the values shown. As we have $L(u_k) = 0$ and $L_c = 2$, these metrics are again given by the cross-correlation of the expected and received channel sequences. Notice, however, that because of the puncturing used the metric values shown in Figure 3.16 are not the same as those in Figure 3.15.

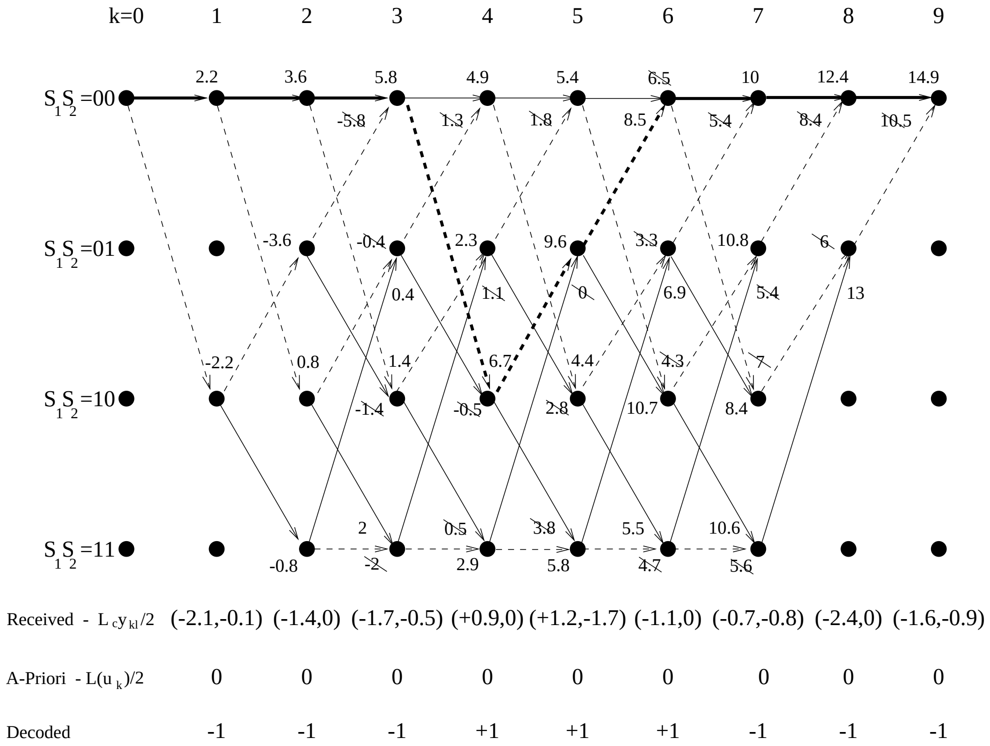


Figure 3.16: Trellis diagram for the SOVA decoding in the first iteration of the first decoder.

To elaborate a little further, the metric calculation philosophy of the SOVA is also different from that of the Viterbi algorithm, since we may have more than one emerging branch at a given state in the trellis. For example, at trellis stage k=5, state S_1S_2=01, both of the two paths emerging from this state survive. This is because these two paths lead to different states at the next stage in the trellis, where both of them win the metric comparison. More specifically, the path associated with the input bit 0 and hence the continuous line leads to state S_1S_2=10 and has a total path metric of 10.7. This metric is compared to the metric of the other path leading to the state S_1S_2=10, which is 4.3 for the path arriving from state S_1S_2=00. Hence the path arriving from the state S_1S_2=01 has the higher metric and therefore it is chosen as the winning path at the new state of S_1S_2=10. Similarly, the other path emerging from the state of S_1S_2=01 at stage k=5 survives. Specifically, this path associated with the input bit of 1 and hence the broken line arrives at state S_1S_2=00 and has a total path metric of 8.5. This is compared to 6.5, namely to the metric of the other path arriving at state S_1S_2=00 and hence this path is selected as the winner.

Again, owing to the puncturing employed the metric values shown in Figure 3.16 are not the same as those in the Viterbi decoding example of Figure 3.15. Despite this the ML path, shown by the bold line in Figure 3.16, is the same as the one that was chosen by the Viterbi algorithm shown in Figure 3.15, with three of the input bits being decoded as +1s rather than -1s.

We now discuss how, having determined the ML path, the SOVA finds the LLRs for the decoded bits. Figure 3.17 is a simplified version of the trellis from Figure 3.16, which shows only the ML path and the paths that merge with this ML path and are discarded. Also shown are the metric differences, denoted by Δ_k^s in Section 3.3.6, between the ML and the discarded paths. These metric differences, together with the previously defined update sequences that indicate for which of the bits the survivor and discarded paths would have given different values, are stored by the SOVA for each node at each stage in the trellis. When the ML path has been identified, the algorithm uses these stored values along the ML path to find the LLR

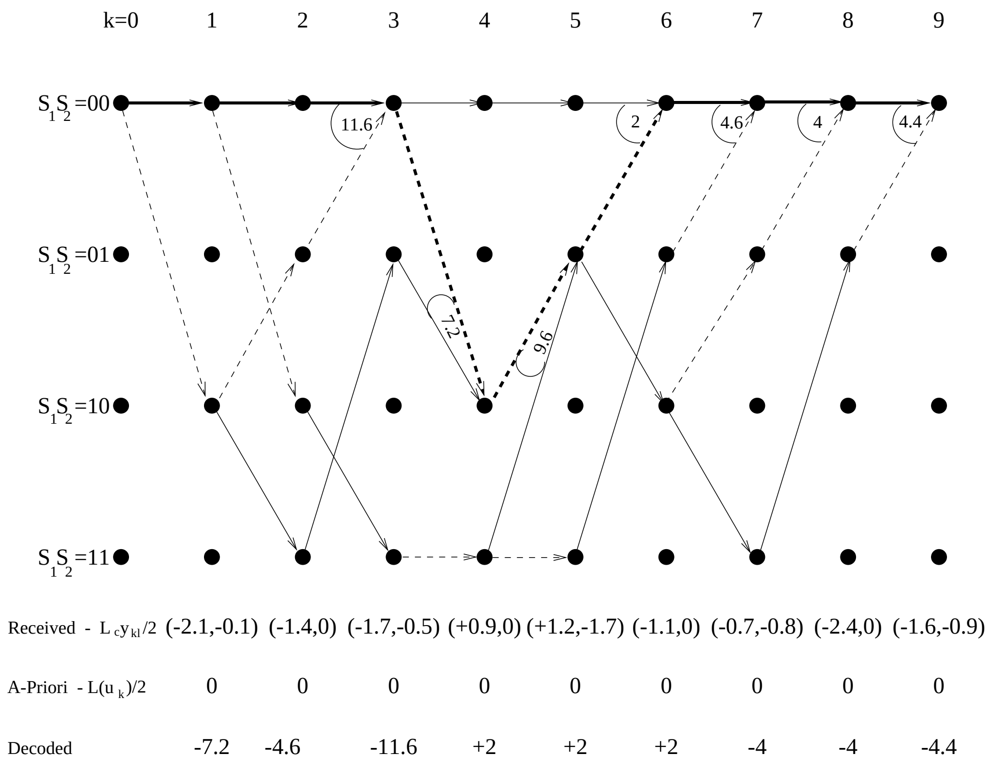


Figure 3.17: Simplified trellis diagram for the SOVA decoding in the first iteration of the first decoder.

Trellis Stage k	Decoded Bit u_k	Metric Difference Δ_k^s	Update Sequence	Decoded LLR
1	-1	–	–	-7.2
2	-1	–	–	-4.6
3	-1	11.6	111	-11.6
4	+1	7.2	1001	+2
5	+1	9.6	10010	+2
6	+1	2	111000	+2
7	-1	4.6	1100010	-4
8	-1	4	11100000	-4
9	-1	4.4	100100000	-4.4

Table 3.2: SOVA output for the first iteration of the first decoder.

for each decoded bit. Table 3.2 shows these stored values for our example trellis shown in Figures 3.16 and 3.17. The calculation of the decoded LLRs shown in this table is detailed at a later stage.

Notice in Table 3.2 that at trellis stages $k = 1$ and $k = 2$ there is no metric difference or update sequence stored because, as can be seen from Figures 3.16 and 3.17, there are no paths merging with the ML path at these stages. For all subsequent stages there is a merging path, and values of the metric differences and update sequences are stored. For the update sequence a 1 indicates that the ML and the discarded merging path would have given different values for a particular bit. At stage k in the trellis we have taken the Most Significant Bit (MSB), on the left-hand side, to represent u_k, the next bit to represent u_{k-1}, etc., until the Least Significant Bit (LSB), which represents u_1. For our RSC code any two paths merging at trellis stage

k give different values for the bit u_k, and so the MSB in the update sequences in Table 3.2 is always 1. Notice furthermore that although in our example the update sequences are all of different lengths, this is only because of the very short frame length we have used. More generally, as explained in Section 3.3.6, all the stored update sequences will be $\delta + 1$ bits long, where δ is usually set to be five times the constraint length of the convolutional code (15 in our case). At this stage it is beneficial for the reader to verify the update sequences of Table 3.2 using Figure 3.17.

We now explain how the SOVA can use the stored update sequences and metric differences along the ML path to calculate the LLRs for the decoded bits. Equation 3.68, which is repeated here as Equation 3.70 for convenience, shows that the decoded a-posteriori LLR $L(u_k|\underline{y})$ for a bit u_k is given by the minimum metric difference of merging paths along the ML path:

$$L(u_k|\underline{y}) \approx u_k \min_{\substack{i=k\cdots k+\delta \\ u_k \neq u_k^i}} \Delta_i^{s_i}. \tag{3.70}$$

This minimum is taken only over the metric differences for stages $i = k, k+1, \ldots, k+\delta$ where the value u_k^i of the bit u_k given by the path merging with the ML path at stage i is different from the value given for this bit by the ML path. Whether or not the condition $u_k = u_k^i$ is met is determined using the stored update sequences. Denoting the update sequence stored at stage l along the ML path as $\underline{e}_l$, for each bit u_k the SOVA examines the MSB of $\underline{e}_k$, the second MSB of $\underline{e}_{k+1}$ etc. up to the $(\delta+1)$th bit (which will be the LSB) of $\underline{e}_{k+\delta}$. For our example this examination of the update sequences is limited because of our short frame length, but the same principles are used. Taking the fourth bit u_4 as an example, to determine the decoded LLR $L(u_4|\underline{y})$ for this bit the algorithm examines the MSB of $\underline{e}_4$ in row 4 of Table 3.2, the second MSB of $\underline{e}_5$ in row 5, etc. up to the sixth MSB of $\underline{e}_9$ in row nine. It can be seen, from the corresponding rows in Table 3.2, that only the paths merging at stages $k = 4$ and $k = 6$ of the trellis give values different from the ML path for the bit u_4. Hence the decoded LLR $L(u_4|\underline{y})$ from the SOVA for this bit is calculated using Equation 3.70 as the value of the bit given by the ML path (+1) times the minimum of the metric differences stored at stages 4 and 6 of the trellis (7.2 and 2), yeilding $L(u_4|\underline{y}) = +2$. For the next bit, u_5, the MSB of $\underline{e}_5$ in row 5 of Table 3.2, the second MSB of $\underline{e}_6$ in row 6, etc. up to the fifth MSB of $\underline{e}_9$ in row 9 are examined, which indicate that a different value for this bit would have been given by the discarded paths at stages 5 and 6 of the trellis. Hence $L(u_5|\underline{y})$ also equals +2, as the metric difference, 2, at stage 6 in the trellis is less than the metric difference, 9.6, at stage 5. For the next bit, u_6, the update sequences indicate that all the merging paths from the sixth stage of the trellis to the end of the trellis would give values different to those given by the ML path. However, the minimum of all these metric differences is still 2, and so $L(u_6|\underline{y})$ also equals +2. This illustrates in the SOVA how one discarded path having a low metric difference can entirely determine the LLRs for all the bits, for which it gives a different value from the ML path, which is a consequence of taking the minimum in Equation 3.70. At stage 6 of the trellis, where the algorithm incorrectly chooses the non-zero path as the survivor, the metric difference between the chosen (incorrect) path and the discarded path is the lowest metric difference (2) encountered along the ML path. Hence the LLRs for the three incorrectly decoded bits, i.e. for u_4, u_5 and u_6, have the lowest magnitudes of any of the decoded bits.

The remaining decoded LLR values in Table 3.2 are computed following a similar procedure. However, also worth noting explicitly is the LLR for the bit u_3. Examination of the MSB of $\underline{e}_3$ in row 3 of Table 3.2, the second MSB of $\underline{e}_4$ in row 4, etc., up to the seventh MSB of $\underline{e}_9$, reveals that only the path merging with the ML path at the third stage of the trellis would give a different value for u_3. Hence the minimum for the LLR $L(u_3|\underline{y})$ in Equation 3.70 is over one term, and the magnitude of $L(u_3|\underline{y})$ is determined by the metric difference of the merging path at stage $k = 3$ of the trellis, which is 11.6.

We now move on to describe the operation of the second component decoder in the first iteration. This decoder uses the extrinsic information from the first decoder as a-priori information to assist its operation, and therefore should be able to provide a better estimate of the encoded sequence than the first decoder was. Equation 3.46 from Section 3.3.4 gives the extrinisic information from the MAP decoder as:

$$L_e(u_k) = L(u_k|\underline{y}) - L(u_k) - L_c y_{ks}. \tag{3.71}$$

The same equation can be derived for all the soft-in soft-out decoders which are used as component decoders for turbo codes. This equation states that the extrinsic information $L_e(u_k)$ is given by the soft output

Trellis Stage k	Decoder LLR Output $L(u_k\|\underline{y})$	A-Priori Info. $L(u_k)$	Received Sys. Info. $L_c y_{ks}$	Extrinsic Information
1	-7.2	0	-4.2	-3
2	-4.6	0	-2.8	-1.8
3	-11.6	0	-3.4	-8.2
4	+2	0	+1.8	+0.2
5	+2	0	+2.4	-0.4
6	+2	0	-2.2	+4.2
7	-4	0	-1.4	-2.6
8	-4	0	-4.8	+0.8
9	-4.4	0	-3.2	-1.2

Table 3.3: Calculation of the extrinsic information from the first decoder in the first iteration using Equation 3.71.

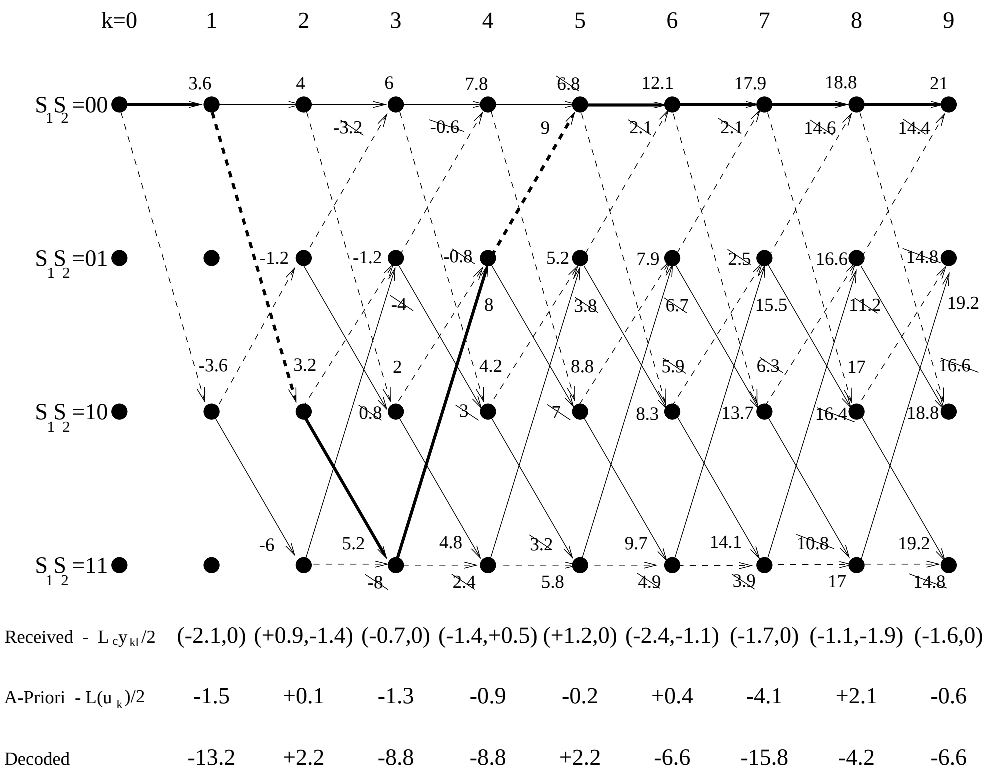


Figure 3.18: Trellis diagram for the SOVA decoding in the first iteration of the second decoder.

$L(u_k|\underline{y})$ from the decoder with the a-priori information $L(u_k)$ (if any was available) and the received systematic channel information $L_c y_{ks}$ subtracted. Table 3.3 shows the extrinsic information calculated from Equation 3.71 from the first decoder, which is then interleaved by a 3×3 block interleaver and used as the a-priori information for the second component decoder. The second component decoder also uses the interleaved received systematic channel values, and the received parity bits from the second encoder which were not punctured (i.e. the second, fourth, sixth and eighth bits).

Figure 3.18 shows the trellis for the SOVA decoding of the second decoder in the first iteration. The extrinsic information values from Table 3.3 are shown after being interleaved and divided by two as $L(u_k)/2$.

Trellis Stage k	Decoder LLR Output $L(u_k\|\underline{y})$	A-Priori Info. $L(u_k)$	Received Sys. Info. $L_c y_{ks}$	Extrinsic Information
1	-13.2	-3	-4.2	-6
2	+2.2	+0.2	+1.8	+0.2
3	-8.8	-2.6	-1.4	-4.8
4	-8.8	-1.8	-2.8	-4.2
5	+2.2	-0.4	+2.4	+0.2
6	-6.6	+0.8	-4.8	-2.6
7	-15.8	-8.2	-3.4	-4.2
8	-4.2	+4.2	-2.2	-6.2
9	-6.6	-1.2	-3.2	-2.2

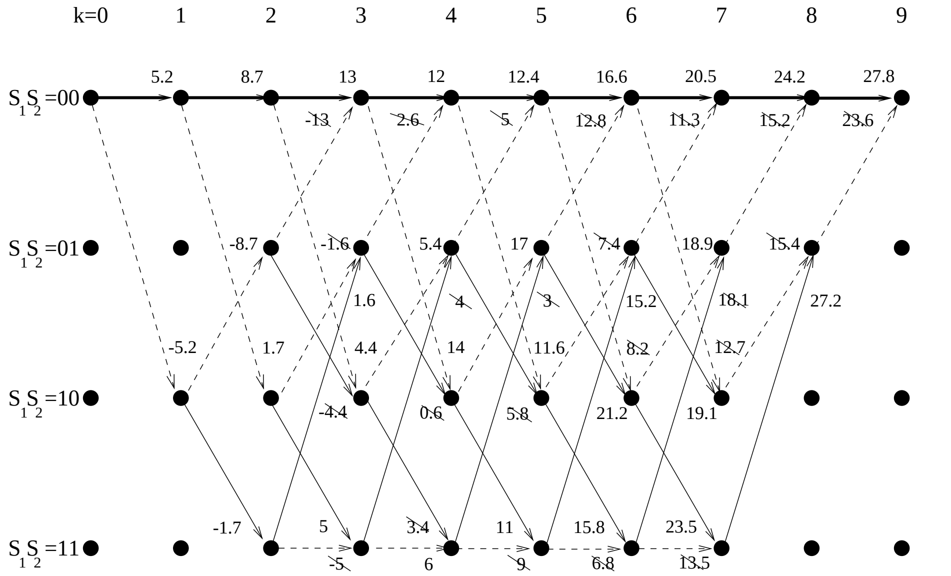
Table 3.4: Calculation of the extrinsic information from the second decoder in the first iteration using Equation 3.71.

Also shown is the channel information $L_c y_{ks}/2$ used by this decoder. Notice that as the trellis is not terminated for the second component encoder, paths terminating in all four possible states of the trellis are considered at the decoder. However, the metric for the $S_1 S_2 = 00$ state is the maximum of the four final metrics, and hence this all-zero state is used as the final state of the trellis. Note furthermore that the metrics in Figure 3.18 are now calculated as the cross-correlation of the received and expected channel information, plus the a-priori information $u_k L(u_k)/2$.

The ML path chosen by the second component decoder is shown by a bold line in Figure 3.18, together with the LLR values output by the decoder. These are calculated, using update sequences and minumum metric differences, in the same way as was explained for the first decoder using Figure 3.17 and Table 3.2. It can be seen that the decoder makes an incorrect decision at stage $k = 5$ in the trellis and selects a path other than the all-zero path as the survivor. However, the incorrectly chosen path gives decoded bits of +1 for only two transitions, and hence only two, rather than three, decoding errors are made. Furthermore the difference in the metrics between the correct and the chosen path at trellis stage $k = 5$ is only 2.2, and so the magnitude of the decoded LLRs $L(u_k|\underline{y})$ for the two incorrectly decoded bits, u_2 and u_5, is only 2.2. This is significantly lower than the magnitudes of the LLRs for the other bits, and indicates that the algorithm is less certain about these two bits being +1 than it is about the other bits being -1.

Having calculated the LLRs from the second component decoder, the turbo decoder has now completed one iteration. The soft-output LLR values from the second component decoder shown in the bottom line of Figure 3.18 could now be de-interleaved and used as the output from the turbo decoder. This de-interleaving would result in an output sequence which gave negative LLRs for all the decoded bits except u_4 and u_5, which would be incorrectly decoded as +1s as their LLRs are both +2.2. Thus, even after only one iteration, the turbo decoder has decoded the received sequence with one less error than the convolutional decoder did. However, generally better results are achieved with more iterations, and so we now progress to describe the operation of the turbo decoder in the second iteration.

In the second, and all subsequent, iterations the first component decoder is able to use the extrinsic information from the second decoder in the previous iteration as a-priori information. Table 3.4 shows the calculation of this extrinsic information using Equation 3.71 from the second decoder in the first iteration. It can be seen that it gives negative LLRs for all the bits except u_2 and u_5, and for these two bits the LLRs are close to zero. This extrinsic information is then de-interleaved and used as the a-priori information for the first decoder in the next (second) iteration. The trellis for this decoder is shown in Figure 3.19. It can be seen that this decoder uses the same channel information as it did in the first iteration. However, now, in contrast to Figure 3.16, it also has a-priori information, to assist it in finding the correct path through the trellis. The selected ML path is again shown by a bold line, and it can be seen that now the correct all-zero path is chosen. The second iteration is then completed by finding the extrinsic information from the first decoder, interleaving it and using it as a-priori information for the second decoder. It can be shown that this decoder will also now select the all-zero path as the ML path, and hence the output from the turbo decoder after the seond iteration will be the correct all -1 sequence. This concludes our example of the operation



Received - $L_c y_{kl}/2$	(-2.1,-0.1)	(-1.4,0)	(-1.7,-0.5)	(+0.9,0)	(+1.2,-1.7)	(-1.1,0)	(-0.7,-0.8)	(-2.4,0)	(-1.6,-0.9)
A-Priori - $L(u_k)/2$	-3	-2.1	-2.1	+0.1	+0.1	-3.1	-2.4	-1.3	-1.1
Decoded	-9.4	-26	-7.4	-3.8	-3.8	-3.8	-9	-9	-4.2

Figure 3.19: Trellis diagram for the SOVA decoding in the second iteration of the first decoder.

of an iterative turbo decoder using the SOVA.

3.3.8 Comparison of the Component Decoder Algorithms

In this section we have detailed the iterative structure and the component decoders used for decoding turbo codes. A numerical example illustrating this decoding was given in the previous section. We now conclude by summarising the operation of the algorithms which can be used as component decoders, highlighting the similarities and differences between these algorithms, and noting their relative complexities and performances.

The MAP algorithm is the optimal component decoder for turbo codes. It finds the probability of each bit u_k being a +1 or -1 by calculating the probability for each transition from state $S_{k-1} = \grave{s}$ to $S_k = s$ that could occur if the input bit was +1, and similarly for every transition that could occur if the input bit was -1. As these transitions are mutually exclusive, the probability of any one of them occurring is simply the sum of their individual probabilities, and hence the LLR for a bit u_k is given by the ratio of two sums of probabilities, as in Equation 3.17.

Owing to the Markov nature of the trellis and the assumption that the output from the trellis is observed in memoryless noise, the individual probabilities of the transitions in Equation 3.17 can be expressed as the product of three terms $\alpha_{k-1}(\grave{s})$, $\beta_k s$ and $\gamma_k(\grave{s}, s)$, as in Equation 3.20. By definition the $\alpha_{k-1}(\grave{s})$ term gives the probability that the trellis reaches state $S_{k-1} = \grave{s}$ and that the received sequence up to this point is $\underline{y}_{j<k}$. The state transition probability, $\gamma_k(\grave{s}, s)$, is defined as the probability that given the trellis is in state $S_{k-1} = \grave{s}$ it moves to state $S_k = s$ and the received sequence for that transition is $\underline{y}_k$. Finally, the $\beta_k(s)$ term gives the probability that given the trellis is in state $S_k = s$, the received sequence from this point to the end of the trellis is $\underline{y}_{j>k}$. The state transition probabilities $\gamma_k(\grave{s}, s)$ are calculated from the received and expected channel sequences, y_{kl} and x_{kl}, for a given transition and the a-priori LLR $L(u_k)$ of the bit associated with this transition, as seen in Equation 3.40 for an AWGN channel. The $\alpha_{k-1}(\grave{s})$ terms can then be calculated using a forward recursion through the trellis, as in Equation 3.25, and similarly the $\beta_k(s)$ terms are calculated using Equation 3.28 by a backward recursion through the trellis. The output LLR $L(u_k|\underline{y})$ from the MAP algorithm is then determined for each bit u_k by finding the probability of each transition from stage S_{k-1} to S_k in the trellis. These transitions are divided into two groups:- those that would have resulted if the bit u_k was a +1, and those that would have resulted if the bit u_k was a -1. The LLR $L(u_k|\underline{y})$ for the bit u_k is then given by the logarithm of the ratio of these probabilities.

The MAP algorithm is optimal for the decoding of turbo codes, but is extremely complex. Furthermore, because of the multiplications used in the recursive calculation of the $\alpha_{k-1}(\grave{s})$ and $\beta_k(s)$ terms, and the exponents used to calculate the $\gamma_k(\grave{s}, s)$ terms, it often suffers from numerical problems in practice. The Log-MAP algorithm is theoretically identical to the MAP algorithm, but transfers its operations to the log domain. Thus multiplications are replaced by additions, and hence the numerical problems of the MAP algorithm are circumvented, while the associated complexity is dramatically reduced.

The Max-Log-MAP algorithm further reduces the complexity of the Log-MAP algorithm using the maximisation approximation given in Equation 3.48. This has two effects on the operation of the algorithm compared to that of the Log-MAP algorithm. First, as can be seen by examining Equation 3.55, it means that only two transitions are considered when finding the LLR $L(u_k|\underline{y})$ for each bit u_k — the best transition from $S_{k-1} = \grave{s}$ to $S_k = s$ that would give $u_k = +1$ and the best that would give $u_k = -1$. Similarly in the recursive calculations of the $A_k(s) = \ln(\alpha_k(s))$ and $B_k(s) = \ln(\beta_k(s))$ terms of Equations 3.52 and 3.53 the approximation means that only one transition, the most likely one, is considered when calculating $A_k(s)$ from the $A_{k-1}(\grave{s})$ terms and $B_{k-1}(\grave{s})$ from the $B_k(s)$ terms. This means that although $A_{k-1}(\grave{s})$ should give the logarithm of the probability that the trellis reaches state $S_{k-1} = \grave{s}$ along *any* path from the initial state $S_0 = 0$, in fact it gives the logarithm of the probability of only the *most likely* path to state $S_{k-1} = \grave{s}$. Similarly $B_k(s) = \ln(\beta_k(s))$ should give the logarithm of the probability of the received sequence $\underline{y}_{j>k}$ given only that the trellis is in state $S_k = s$ at stage k. However, the maximisation in Equation 3.53 used in the recursive calculation of the $B_k(s)$ terms means that only the most likely path from state $S_k = s$ to the end of the trellis is considered, and not all paths.

Hence the Max-Log-MAP algorithm finds the LLR $L(u_k|\underline{y})$ for a given bit u_k by comparing the probability of the most likely path giving $u_k = +1$ to the probability of the most likely path giving $u_k = -1$. For the next bit, u_{k+1}, again the best path that would give $u_{k+1} = +1$ and the best path that would give $u_{k+1} = -1$ are compared. One of these 'best paths' will always be the ML path, and so will not change from one stage to the next, whereas the other may change. In contrast the MAP and the Log-MAP algorithms consider every path in the calculation of the LLR for each bit. All that changes from one stage to the next is the division of paths into those that give $u_k = +1$ and those that give $u_k = -1$. Thus the Max-Log-MAP algorithm gives a degraded performance compared to the MAP and Log-MAP algorithms.

In the SOVA the ML path is found by maximising the metric given in Equation 3.63. The recursion used to find this metric is identical to that used to find the $A_k(s)$ terms in Equation 3.52 in the Max-Log-MAP algorithm. Once the ML path has been found, the hard decision for a given bit u_k is determined by which transition the ML path took between trellis stages S_{k-1} and S_k. The LLR $L(u_k|\underline{y})$ for this bit is determined by examining the paths which merge with the ML path that would have given a different hard decision for the bit u_k. The LLR is taken to be the minimum metric difference for these merging paths which would have given a different hard decision for the bit u_k. Using the notation associated with the Max-Log-MAP algorithm, once a path merges with the ML path, it will have the same value of $B_k(s)$ as the ML path. Hence, as the metric in the SOVA is identical to the $A_k(s)$ values in the Max-Log-MAP algorithm, taking the difference between the metrics of the two merging paths in the SOVA is equivalent to taking the difference between two values of $(A_{k-1}(\grave{s}) + \Gamma_k(\grave{s}, s) + B_k(s))$ in the Max-Log-MAP algorithm, as in Equation 3.55. The only difference is that in the Max-Log-MAP algorithm one path will be the ML path, and the other will be the most likely path that gives a different hard decision for u_k. In the SOVA again one path will be the ML path, but the other may not be the most likely path that gives a different hard decision for u_k. Instead, it will be the most likely path that gives a different hard decision for u_k and survives to merge with the ML path. Other, more likely paths, which give a different hard decision for the

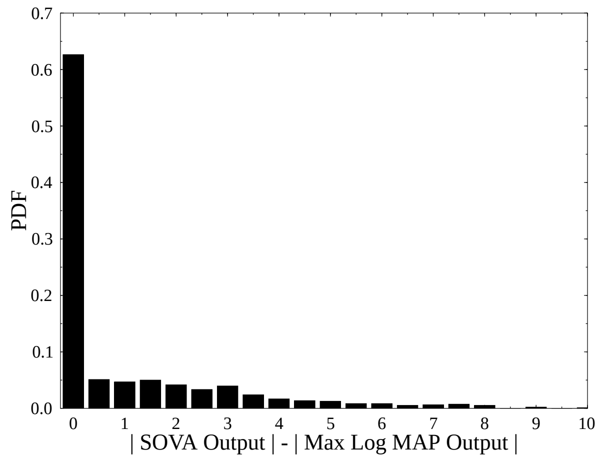


Figure 3.20: Probability density function of differences between absolute values of soft outputs from the SOVA and Max-Log-MAP algorithms.

bit u_k to the ML path, may have been discarded before they merge with the ML path. Thus the SOVA gives a degraded performance compared to the Max-Log-MAP algorithm. However, as pointed out in [52] by Robertson *et al.*, the SOVA and Max-Log-MAP algorithms will always give the same hard decisions, as in both algorithms these hard decisions are determined by the ML path, which is calculated using the same metric in both algorithms.

It was also noted in [52] that the outputs from the SOVA contain no bias when compared to those from the Max-Log-MAP algorithm, they are just more noisy. However, consideration of the arguments above makes it clear that when the most likely path that gives a different hard decision for u_k survives to merge with the ML path, the outputs from the SOVA and the Max-Log-MAP algorithms will be identical. Otherwise when this most likely path which gives a different hard decision for u_k does not survive to merge with the ML path, the merging path that is used to calculate the soft output from the SOVA will be less likely than the path which should have been used. Thus it will have a lower metric, and so the metric difference used in Equation 3.68 will be higher than it should be. Therefore, although the sign of the soft outputs from the SOVA will be identical to those from the Max-Log-MAP algorithm, their magnitudes will be either identical or higher. This can be seen in Figure 3.20, which shows the Probability Density Function (PDF) for the differences between the absolute values of the soft outputs from the SOVA and the Max-Log-MAP algorithms. Both decoders were used as the first decoder in the first iteration, and again the same encoder, all -1 input sequence, and channel SNR as described for Figure 3.11 were used. It can be seen that for more than half of the decoder outputs the two algorithms give identical values. It can also be seen that the absolute values given by the SOVA are never less than those given by the Max-Log-MAP algorithm.

A comparison of the complexities of the Log-MAP, the Max-Log-MAP and the SOVA algorithms is given in [52]. The relative complexity of the algorithms depends on the constraint length K of the convolutional codes used, but it is shown that the Max-Log-MAP algorithm is about twice as complex as the SOVA. The Log-MAP algorithm is about 50% more complex than the Max-Log-MAP algorithm owing to the look-up operation required for finding the correction factors $f_c(x)$. Viterbi noted furthermore [154,155] that the Max-Log-MAP algorithm can be viewed as two generalised Viterbi decoders (one for the forward recursion, and one for the backward recursion) together with a generalised dual-maxima computation (for calculating the soft outputs using Equation 3.55). The complexity of the dual-maxima computation is lower than that of the Viterbi decoder, and hence Viterbi estimated the complexity of the Max-Log-MAP algorithm to be lower than three times that of the Viterbi algorithm. Our related calculations suggest for a $K = 3$ code that the Max-Log-MAP decoder is about 2.6 times as complex, and the Log-MAP decoder is about four times as complex, as the standard Viterbi algorithm.

The performance of the algorithms when used in the iterative decoding of turbo codes follows in the same order as their complexities, with the best performance given by the Log-MAP algorithm, then the Max-Log-MAP algorithm, and the worst performance exhibited by the SOVA. We will compare the performance of these three algorithms when decoding various turbo codes in Section 3.4.

3.3.9 Conclusions

In this section we have described the techniques used for the decoding of turbo codes. Although it is possible to optimally decode turbo codes in a single non-iterative step, for complexity reasons a non-optimum iterative decoder is almost always preferred. Such an iterative decoder employs two component soft-in soft-out decoders, and we have described the MAP, Log-MAP, Max-Log-MAP and SOVA algorithms, which can all be used as the component decoders. The MAP algorithm is optimal for this task, but it is extremely complex. The Log-MAP algorithm is a simplification of the MAP algorithm, and offers the same optimal performance with a reasonable complexity. The other two algorithms, the Max-Log-MAP and the SOVA, are both less complex again, but give a slightly degraded performance.

Having described the principles behind the encoding and decoding of turbo codes we now move on to present our simulation results demonstrating the excellent performance of turbo codes for various scenarios.

3.4 Turbo-coded BPSK Performance over Gaussian Channels

In the previous two sections we have discussed the structure of both the encoder and the decoder in a turbo codec. In this section we present simulation results for turbo codes using BPSK over AWGN channels. We show that there are many parameters, some of which are interlinked, which affect the performance of turbo codes. Some of these parameters are:

- The component decoding algorithm used.
- The number of decoding iterations used.
- The frame length or latency of the input data.
- The specific design of the interleaver used.
- The generator polynomials and constraint lengths of the component codes.

In this section we investigate how all of these parameters affect the performance of turbo codes. The standard parameters we have used in our simulations are shown in Table 3.5. All our results presented in this section consider turbo codes using BPSK modulation over an AWGN channel. The turbo encoder uses two component RS codes in parallel. Our standard RSC component codes are $K = 3$ codes with generator polynomials $G_0 = 7$ and $G_1 = 5$ in octal representation. These generator polynomials are optimum in terms of maximising the minimum free distance of the component codes [126]. The effects of varying these generator polynomials are examined in Section 3.4.5. The standard interleaver used between the two component RSC codes is a 1000-bit random interleaver with odd–even separation [68]. The effects of changing the length of the interleaver, and its structure, are examined in Sections 3.4.4 and 3.4.6. Unless otherwise stated, the results of this section are valid for half-rate codes, where half the parity bits generated by each of the two component RSC codes are punctured. However, for comparison, we also include some results for turbo codes where all the parity bits from both component encoders are transmitted, leading to a one-third-rate code. At the decoder two component, soft-in soft-out, decoders are used in parallel in the structure shown in Figure 3.3. In most of our simulations we use the Log-MAP decoder, but the effect of using other component decoders is investigated in Section 3.4.3. Usually eight iterations of the component decoders are used, but in the next section we consider the effect of the number of iterations.

Channel	Additive White Gaussian Noise (AWGN)
Modulation	Binary Phase Shift Keying (BPSK)
Component Encoders	Two identical recursive - convolutional codes
RSC Parameters	$n = 2, k = 1, K = 3$ $G_0 = 7, G_1 = 5$
Interleaver	1000-bit random interleaver with odd–even separation [68]
Puncturing Used	Half parity bits from each component encoder transmitted give half-rate code
Component Decoders	Log-MAP decoder
Iterations	8

Table 3.5: Standard turbo encoder and decoder parameters used.

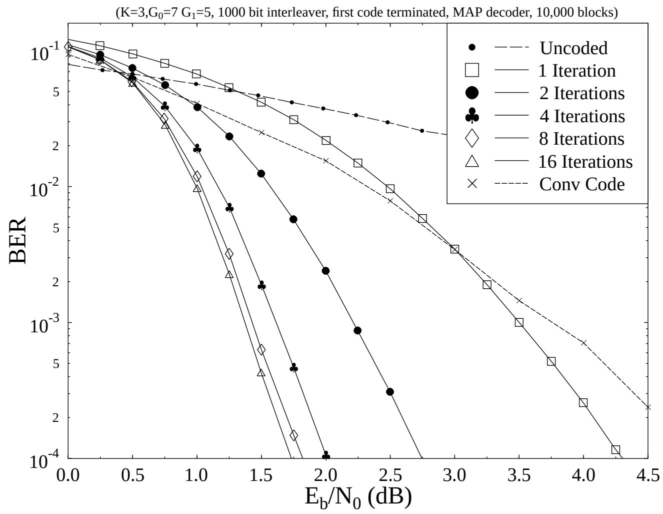


Figure 3.21: Turbo coding BER performance using different numbers of iterations of the MAP algorithm. Other parameters as in Table 3.5.

3.4.1 Effect of the Number of Iterations Used

Figure 3.21 shows the performance of a turbo decoder using the MAP algorithm versus the number of decoding iterations which were used. For comparison, the uncoded BER and the BER obtained using convolutional coding with a standard (2,1,3) non-recursive convolutional code are also shown. Like the component codes in the turbo encoder, the convolutional encoder uses the optimum octal generator polynomials of 7 and 5. It can be seen that the performance of the turbo code after one iteration is roughly similar to that of the convolutional code at low SNRs, but improves more rapidly than that of the convolutional coding as the SNR is increased. As the number of iterations used by the turbo decoder increases, the turbo decoder performs significantly better. However, after eight iterations there is little improvement achieved by using further iterations. For example, it can be seen from Figure 3.21 that using 16 iterations rather than

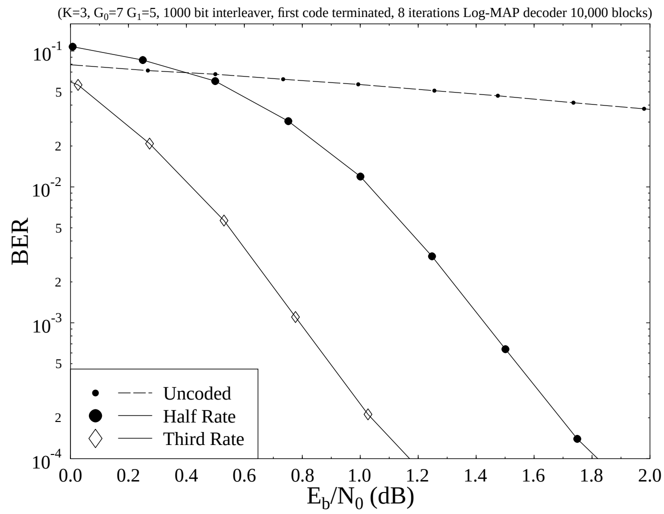


Figure 3.22: BER performance comparison between one-third- and half-rate turbo codes using parameters of Table 3.5.

eight gives an improvement of only about 0.1 dB. Similar results are obtained when using the SOVA; again there is little improvement in the BER performance of the decoder from using more than eight iterations. Hence for complexity reasons usually only between about 4 and 12 iterations are used. Accordingly, unless otherwise stated, in our simulations we used eight iterations. In the next section we consider the effects of puncturing.

3.4.2 Effects of Puncturing

As described in Section 3.2, in a turbo encoder two or more component encoders are used for generating parity information from an input data sequence. In our work we have used two RSC component encoders, and this is the arrangement most commonly used for turbo codes having coding rates below two-thirds. Typically, in order to generate a half-rate code, half the parity bits from each component encoder are punctured. This was the arrangement used in the seminal paper by Berrou *et al.* on the concept of turbo codes [12]. However, it is of course possible to omit the puncturing and transmit all the parity information from both component encoders, which gives a one-third-rate code. The performance of such a code, compared to the corresponding half-rate code, is shown in Figure 3.22. In this figure the encoders use the same parameters as were described above for Figure 3.21. It can be seen that transmitting all the parity information gives a gain of about 0.6 dB, in terms of E_b/N_0, at a BER of 10^{-4}. This corresponds to a gain of about 2.4 dB in terms of channel SNR. Very similar gains are seen for turbo codes with different frame lengths. Let us now consider the performance of the various soft-in soft-out component decoding algorithms which were described in Section 3.3.

3.4.3 Effect of the Component Decoder Used

Figure 3.23 shows a comparison between turbo decoders using the different component decoders described in Section 3.3 for a turbo encoder using the parameters described above. In this figure the 'Log MAP (Exact)' curve refers to a decoder which calculates the correction term $f_c(x)$ in Equation 3.56 of Section 3.3.5

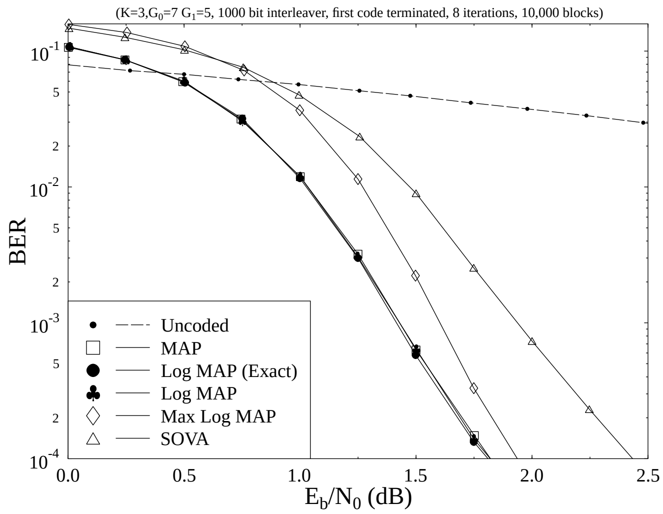


Figure 3.23: BER performance comparison between different component decoders for a random interleaver with L=1000. Other parameters as in Table 3.5.

exactly, i.e. using:

$$f_c(x) = \ln(1 + e^{-x}), \tag{3.72}$$

rather than using a look-up table as described in [52]. The Log-MAP curve refers to a decoder which does use a look-up table with eight values of $f_c(x)$ stored, and hence introduces an approximation to the calculation of the LLRs. It can be seen that, as expected, the MAP and the Log-MAP (exact) algorithms give identical performances. Furthermore, as Robertson *et al.* found [52], the look-up procedure for the values of the $f_c(x)$ correction terms introduces no degradation to the performance of the decoder.

It can also be seen from Figure 3.23 that the Max-Log-MAP and the SOVA both give a degradation in performance compared to the MAP and Log-MAP algorithms. At a BER of 10^{-4} this degradation is about 0.1 dB for the Max-Log-MAP algorithm, and about 0.6 dB for the SOVA.

Figure 3.24 compares the Log-MAP, Max-Log-MAP and SOVA algorithms for a turbo decoder with a frame length of only 169 bits, rather than 1000 bits as was used for Figure 3.23. It can be seen that although all three decoders give a worse BER performance than those shown in Figure 3.23, the differences in the performances between the decoders are very similar to those shown in Figure 3.23. Similarly, Figure 3.25 compares these three decoding algorithms for a one-third-rate code, and again the degradations relative to a decoder using the Log-MAP algorithm are about 0.1 dB for the Max-Log-MAP algorithm, and about 0.6 dB for the SOVA.

3.4.4 Effect of the Frame Length of the Code

In the original paper on turbo coding by Berrou *et al.* [12], and in many of the subsequent papers, impressive results have been presented for coding with very large frame lengths. However, for many applications, such as for example speech transmission systems, the large delays inherent in using high frame lengths are unacceptable. Therefore an important area of turbo coding research is achieving as impressive results with short frame lengths as have been demonstrated for long frame length systems.

Figure 3.26 shows how dramatically the performance of turbo codes depends on the frame length L used

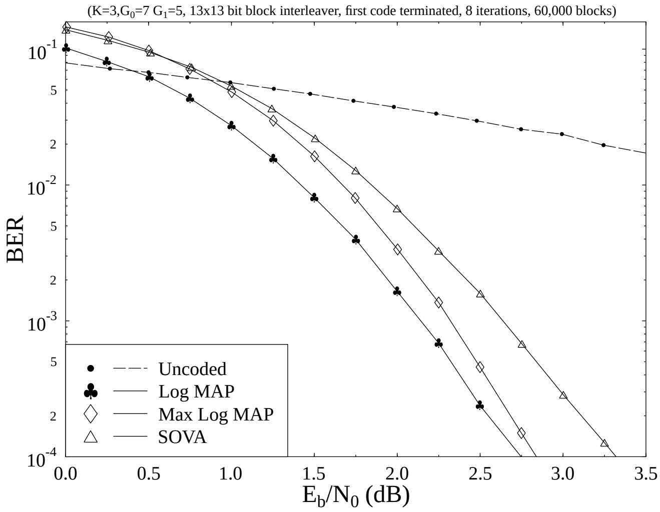


Figure 3.24: BER performance comparison between different component decoders for a L=169, 13×13, block interleaver. Other parameters as in Table 3.5.

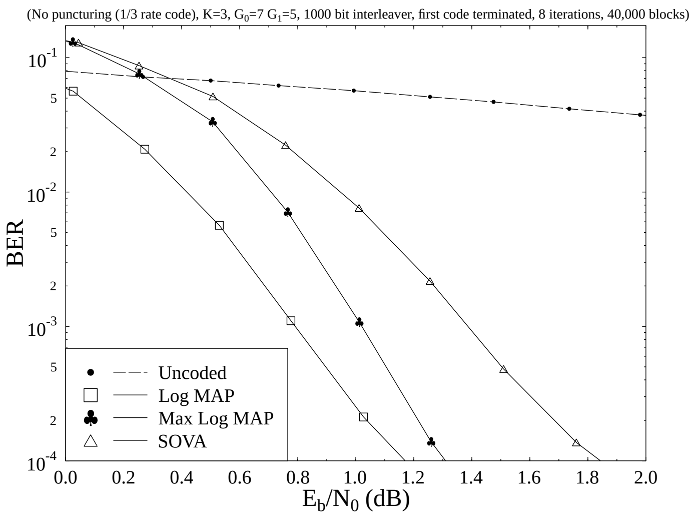


Figure 3.25: BER performance comparison between different component decoders for a random interleaver with L=1000 using a one-third rate code. Other parameters as in Table 3.5.

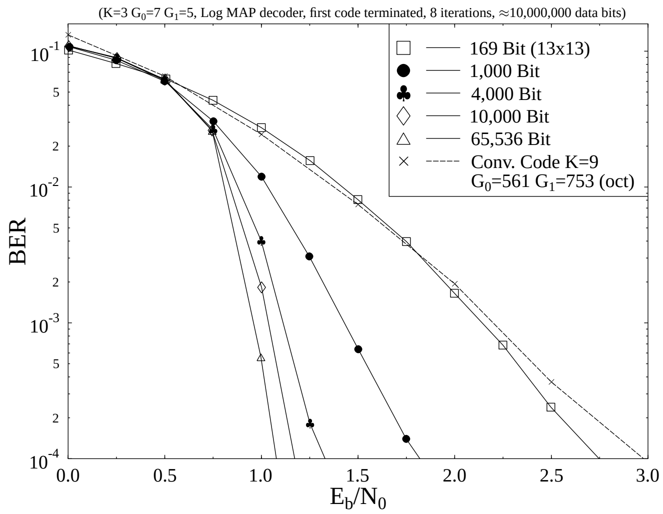


Figure 3.26: Effect of frame length on the BER performance of turbo coding. All interleavers except $L = 169$ block interleaver use random separated Interleavers [68]. Other Parameters as in Table 3.5.

in the encoder. The 169-bit code would be suitable for use in a speech transmission system at approximately 8 kbit/s with a 20 ms frame length [156], while the 1000-bit code would be suitable for video transmission. The larger frame length systems would be useful in data or non-real-time transmission systems. It can be seen from Figure 3.26 that the performance of turbo codes is very impressive for systems with long frame lengths. However, even for a short frame length system, using 169 bits per frame, it can be seen that turbo codes give good results, comparable to or better than a constraint length $K = 9$ convolutional code. The use of the $K = 9$ convolutional code as a bench-marker is justified below.

As noted in Section 3.3, a single decode with the Log-MAP decoder is about four times as complex as decoding the same code using a standard Viterbi decoder. The curves shown in Figure 3.26, and in most of our results, use two component decoders with eight iterations. Therefore the overall complexity of a turbo decoder is approximately $2 \times 8 \times 4 = 64$ times that of a Viterbi decoder for one of the component convolutional codes. This means that the complexity of our turbo decoder using eight iterations of two $K = 3$ component codes is approximately the same as the complexity of a Viterbi decoder for an ordinary $K = 9$ convolutional code. In order to provide a comparison between the performance of turbo codes and convolutional codes for similar complexity decoders, we will compare our $K = 3$ turbo codes with an eight iteration decoder to a $K = 9$ convolutional code.

Figure 3.26 shows the performance of such a convolutional code. A non-recursive (2,1,9) convolutional code using the generator polynomials $G_0 = 561$ and $G_1 = 753$ in octal notation, which maximise the free distance of the code [126], was used. These generator polynomials provide the best performance in the AWGN channels we use in this section. A frame length of 169 bits is used, and the code is terminated. It can be seen that even for the short frame length of 169 bits, turbo codes outperform similar complexity convolutional codes. As the frame length is increased, the performance gain from using turbo codes, rather than high-constraint-length convolutional codes, increases dramatically.

Figure 3.27 shows how the performance of a one-third rate turbo code varies with the frame length of the code. Again, the performance of the turbo code is better the longer the frame length of the code, but impressive results are still obtained with a frame length of only 169 bits. Again the results for a $K = 9$

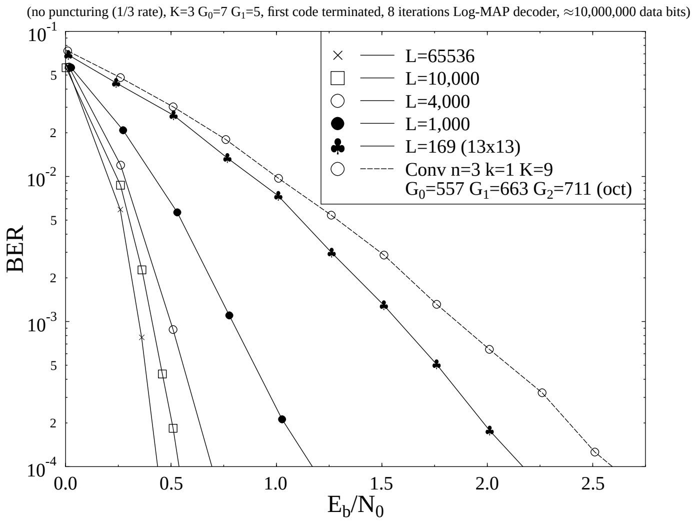


Figure 3.27: Effect of frame length on BER performance of one-third rate turbo coding. All interleavers except $L = 169$ block interleaver use random interleavers. Other parameters as in Table 3.5.

convolutional code are shown, this time using a third-rate $n = 3, k = 1$ code with the optimal generator polynomials of $G_0 = 557$, $G_1 = 663$ and $G_2 = 711$ [126] in octal notation. Again it can be seen that the high-constraint-length convolutional code is outperformed by turbo codes with frame lengths of 169 and higher.

Let us now consider the effect of using different RSC component codes.

3.4.5 The Component Codes

Both the constraint length and the generator polynomials used in the component codes of turbo codes are important parameters. Often in turbo codes the generator polynomials which lead to the largest minimum free distance for ordinary convolutional codes are used, although when the effect of interleaving is considered these generator polynomials do not necessarily lead to the best minimum free distance for turbo codes. Figure 3.28 shows the huge difference in performance that can result from different generator polynomials being used in the component codes. The other parameters used in these simulations were the same as detailed above in Table 3.5.

Most of the results provided in this chapter were obtained using constraint length 3 component codes. For these codes we have used the optimum generator polynomials in terms of maximising the minimum free distance of the component convolutional codes, i.e. 7 and 5 in octal representation. These generator polynomials were also used for constraint length 3 turbo coding by Hagenauer *et al.* in [61] and Jung in [66]. It can be seen from Figure 3.28 that the order of these generator polynomials is important — the value 7 should be used for the feedback generator polynomial in Figure 3.1 (denoted in our work by G_0). If G_0 and G_1 are swapped round, the performance of a convolutional code (both regular and recursive systematic codes) would be unaffected, but for turbo codes this gives a significant degradation in performance.

The effect of increasing the constraint length of the component codes used in turbo codes is shown in Figure 3.29. For the constraint length 4 turbo code we again used the optimum minimum free distance generator polynomials for the component codes (15 and 17 in octal, 13 and 15 in decimal representations). The resulting turbo code gives an improvement of about 0.25 dB at a BER of 10^{-4} over the $K = 3$ curve.

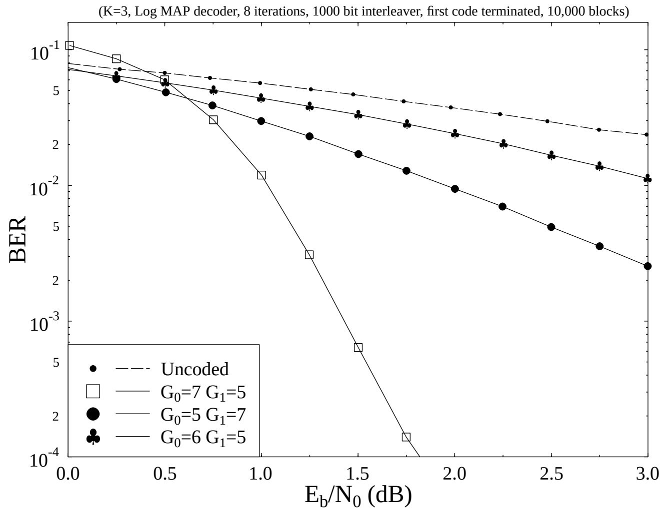


Figure 3.28: Effect of generator polynomials on BER performance of turbo coding. Other parameters as in Table 3.5.

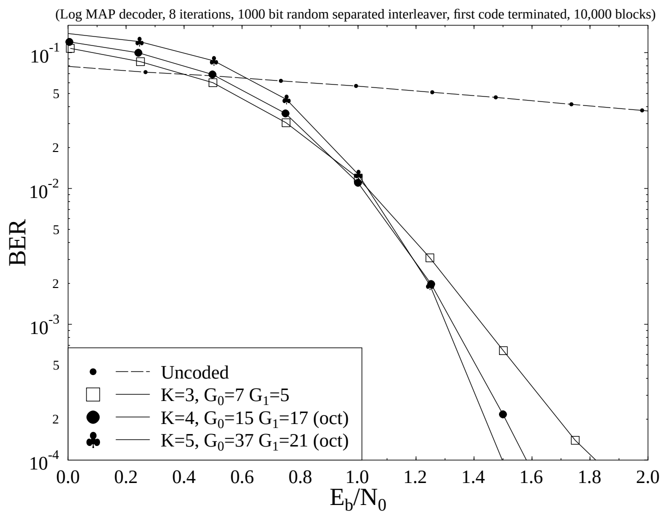


Figure 3.29: Effect of constraint length on the BER performance of turbo coding. Other parameters as in Table 3.5.

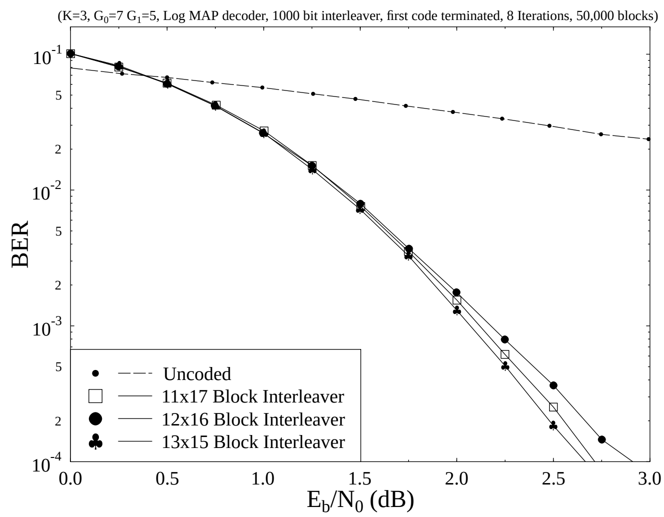


Figure 3.30: Effect of block interleaver choice for $L \approx 190$ frame length turbo codes. Other parameters as in Table 3.5.

For the constraint length 5 turbo code we used the octal generator polynomials 37 and 21 (31 and 17 in decimal), which were the polynomials used by Berrou *et al.* [12] in the original paper on turbo coding. We also tried using the octal generator polynomials 23 and 35 (19 and 29), which are again the optimum minimum free distance generator polynomials for the component codes, as suggested by Hagenauer *et al.* in [61]. We found that these generator polynomials gave almost identical results to those used by Berrou *et al.* It can be seen from Figure 3.29 that increasing the constraint length of the turbo code does improve its performance, with the $K = 4$ code performing about 0.25 dB better than the $K = 3$ code at a BER of 10^{-4}, and the $K = 5$ code giving a further improvement of about 0.1 dB. However, these improvements are provided at the cost of approximately doubling or quadrupling the decoding complexity. Therefore, unless otherwise stated, we have used component codes with a constraint length of 3 in our work. Let us now focus on the effects of the interleaver used within the turbo encoder and decoder.

3.4.6 Effect of the Interleaver

It is well known that the interleaver used in turbo codes has a vital influence on the performance of the code. The interleaver design together with the generator polynomials used in the component codes, and the puncturing used at the encoder, have a dramatic affect on the free distance of the resultant turbo code. Several algorithms have been proposed, for example in references [150] and [65], that attempt to choose good interleavers based on maximising the minimum free distance of the code. However, this process is complex, and the resultant interleavers are not necessarily optimum. For example, in [157] random interleavers designed using the technique given in [65] are compared to a 12×16 dimensional block interleaver, and the 'optimised' interleavers are found to perform worse than the block interleaver.

In [68] a simple technique for designing good interleavers, which is referred to as 'odd–even separation' is proposed. With alternate puncturing of the parity bits from each of the component codes, which is the puncturing most often used, if an interleaver is designed so that the odd and even input bits are kept separate, then it can be shown that one (and only one) parity bit associated with each information bit will be left unpunctured. This is preferable to the more general situation, where some information bits will have their

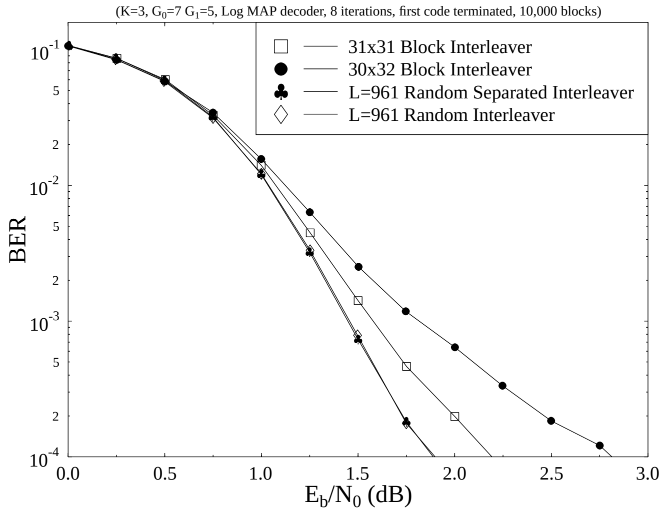


Figure 3.31: Effect of interleaver choice for $L \approx 961$ frame length turbo codes. Other parameters as in Table 3.5.

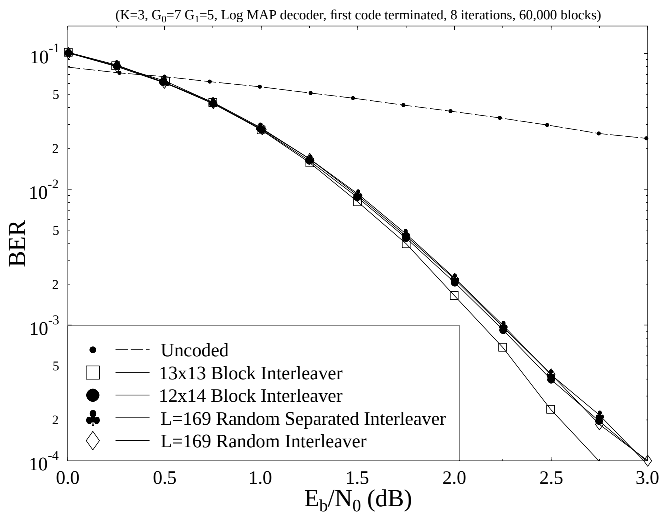


Figure 3.32: Effect of interleaver choice for $L \approx 169$ frame length turbo codes. Other parameters as in Table 3.5.

parity bits from both component codes transmitted, whereas others will have neither of their parity bits transmitted.

A convenient way of achieving odd–even separation in the interleaver is to use a block interleaver with an odd number of rows and columns [68]. The benefits of using an odd number of rows and columns with a block interleaver can be seen in Figure 3.30. This shows a comparison between turbo coders using several block interleavers with frame lengths of approximately 190 bits. The 12×16 dimensional block interleaver, proposed for short frame transmission systems in [157] and used by the same authors in other papers such as [66, 67, 158], clearly has a somewhat lower performance than the other block interleavers, which use an odd number of rows and columns. It is also interesting to note that of the two block interleavers with an odd number of rows and columns, the interleaver which is closer to being square (i.e. the 13×15 interleaver) performs better than the more rectangular 11×17 interleaver.

We also attempted using random interleavers of various frame lengths. The effect of the interleaver choice for a turbo coding system with a frame length of approximately 960 bits is shown in Figure 3.31. It can be seen from this figure that, as was the case with the codes with frame lengths around 192 bits shown in Figure 3.30, the block interleaver with an odd number of rows and columns (the 31×31 interleaver) performs significantly better than the interleaver with an even number of rows and columns (the 30×32 interleaver). However, both of these interleavers are outperformed by the two random interleavers. In the 'random separated' interleaver odd–even separation, as proposed by Barbulescu and Pietrobon [68], is used. This interleaver performs very slightly better than the other random interleaver, which does not use odd–even separation. However, the effect of odd–even separation is much less significant for the random interleavers than it is for the block interleavers.

Similar curves are shown in Figure 3.32 for turbo coding schemes with approximately 169 bits per frame. It can be seen again that the scheme using block interleaving with odd–even separation (i.e. the 13×13 interleaver) performs better than the scheme using block interleaving without odd–even separation (i.e. the 12×14 interleaver). However, for this short frame length system the two random interleavers perform worse than the best block interleaver. From our results it appears that although random interleavers give the best performance for turbo codes with long frame lengths, for short frame length systems the best performance is given using a block interleaver with an odd number of rows and columns.

When puncturing is not used, and we have a third-rate code, the benefit of using odd–even separation with block interleavers, i.e. using block interleavers with an odd number of rows and columns, disappears. This can be seen from Figure 3.33, which compares the performance of a turbo code with no puncturing using three different interleavers, all with a length of approximately 169 bits. As in the case of the half-rate turbo codes using puncturing in Figure 3.32, for a small frame length, such as 169 bits, the best performance is given by using a block rather than a random interleaver. However, it can be seen from Figure 3.33 that, unlike for half-rate codes, for turbo codes without puncturing there is little difference between the block interleavers with and without odd–even separation, i.e. between the 13×13 and 12×14 interleavers.

In [159] Herzberg suggests that a 'reverse block' interleaver, i.e. a block interleaver in which the output bits are read from the block in the reverse order relative to an ordinary block interleaver, gives an improved performance over ordinary block interleavers. He also suggests that for high SNRs, and hence for low BERs, reverse block interleavers having a short frame length give a better performance than random interleavers having a significantly higher frame length. However, as can be seen from Figure 3.34, which portrays the performance of ordinary and reverse block interleavers for various frame lengths, we found very little difference between the performances of block and reverse block interleavers. One difference between our results and those in [159] is that we have used punctured half-rate turbo codes, whereas Herzberg used turbo codes without puncturing. However, we found that even with third-rate turbo codes using no puncturing, and using 14×14 interleavers as Herzberg did, the performances of block and reverse block interleavers were almost identical. It appears in [159] that for turbo codes with long random interleavers, and with an ordinary block interleaver, Herzberg used the generator polynomials $G_0 = 5$ and $G_1 = 7$, whereas for the reverse block interleaver he used the generator polynomials $G_0 = 7$ and $G_1 = 5$. The generator polynomials $G_0 = 5$ and $G_1 = 7$ were used so that the performance of turbo codes with long random interleavers could be approximated using the Union bound and the error coefficients calculated by Benedetto and Montorsi in [59] for these generator polynomials. However, as was seen in Figure 3.28, these generator polynomials give a significantly worse performance than the generator polynomials $G_0 = 7$ and

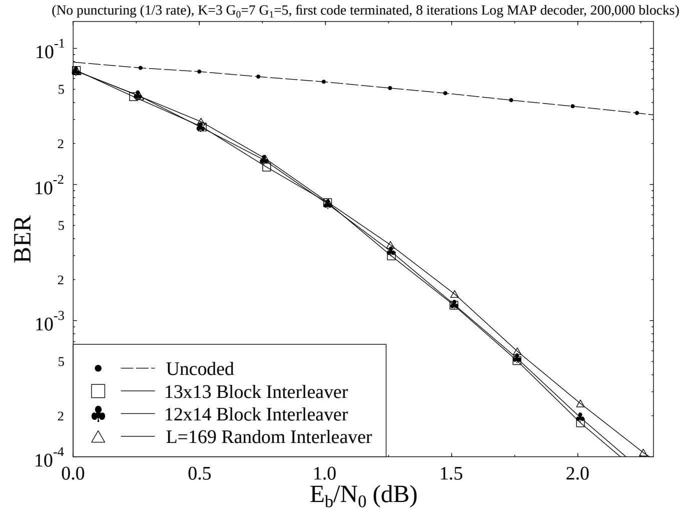


Figure 3.33: Effect of interleaver choice for third-rate $L \approx 169$ frame length turbo codes. Other parameters as in Table 3.5.

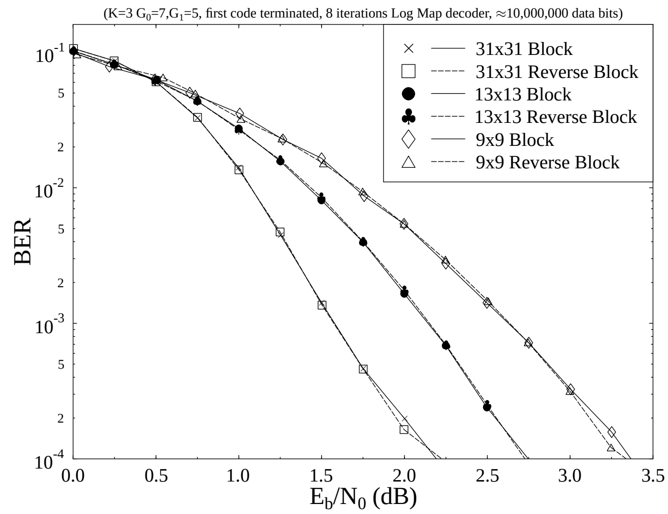


Figure 3.34: BER performance of block and reverse block interleavers. Other parameters as in Table 3.5.

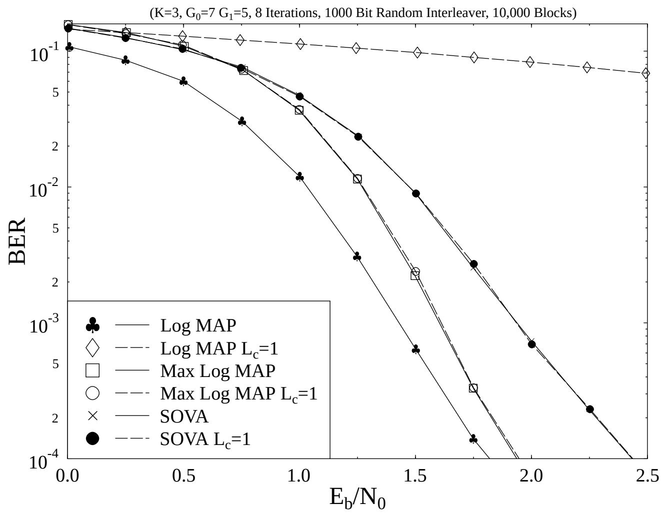


Figure 3.35: Effect of using incorrect channel reliability measures L_c on an iterative turbo decoder using various component decoders. Other parameters as in Table 3.5.

$G_1 = 5$ we have used for most of our simulations, and Herzberg used with his reverse block interleaver. Thus it appears that the reason Herzberg found such promising results for the reverse block interleaver was not because of this interleaver's superiority, but because of the inferiority of the generator polynomials he used with random and block interleavers.

Let us now focus our attention on the effect of the estimation of the channel reliability measure L_c.

3.4.7 Effect of Estimating the Channel Reliability Value L_c

In the previous section we highlighted how the component decoders of an iterative turbo decoder interacted using soft inputs, the channel inputs $L_c y_{kl}$ as well as the a-priori inputs $L(u_k)$, and provided the a-posteriori LLRs $L(u_k|\underline{y})$ as soft outputs. In the MAP and Log-MAP algorithms the channel inputs and a-priori information are used to calculate the transition probabilities $\gamma_k(\grave{s}, s)$ that are then used to recursively calculate the $\alpha_k(s)$ and $\beta_k(s)$ and finally the a-posteriori LLRs $L(u_k|\underline{y})$. Similarly, in the Max-Log-MAP and the SOVA algorithms the channel and a-priori information values are used to update metrics, which are then used to give the soft-output a-posteriori LLRs. In this section we investigate how important an accurate estimate of the channel reliability measure L_c is to the good performance of an iterative turbo decoder.

Figure 3.35 shows the performance of iterative turbo decoders using three different component decoders:- the Log-MAP, Max-Log-MAP and the SOVA algorithms. For each component decoder type the continuous line shows the performance of the codec when the channel reliability value L_c is calculated exactly using the known channel SNR. For all our previous results we assumed that the channel SNR, and hence the correct value of L_c, would be known at the decoder. The broken curves in Figure 3.35 show how the three component decoders perform when L_c is not known. For these curves the value of $L_c = 1$, which corresponds to a value of E_b/N_0 of -3 dB, was used at all channel SNRs. It can be seen from Figure 3.35 that for the SOVA and the Max-Log-MAP algorithms the turbo decoder performs equally well whether or not the correct value of L_c is known. However, for the Log-MAP algorithm the performance of the iterative turbo decoder is drastically affected by the value of L_c used.

The reason for these effects can be understood by considering the different operation of the three algorithms, as was described in Section 3.3. In the SOVA the channel values $L_c y_{kl}$ are used to recursively calculate the metrics M_n using Equation 3.63. The metric M_n for a state $S_k = s$ along a path is given by the metric M_{n-1} for the previous state along the path, added to an a-priori information term and to a cross-correlation term between the expected and the received channel values, x_{kl} and y_{kl}. The channel reliability measure L_c is used to scale this cross-correlation between the received and expected channel values. Then, once the ML path has been identified, the soft outputs from the algorithm are given by the minimum metric difference between the ML path and the paths merging with this ML path, as seen in Equation 3.68. When we use an incorrect value of L_c, effectively we are scaling the inputs to the component decoders by a factor. For instance, if we simply use $L_c = 1$, then we scale the channel input values by a factor of one over the correct value of L_c. In the SOVA this has the effect of scaling all the metrics M_n by the same factor, and as the a-posteriori LLRs from the algorithm are given by the difference between metrics for different paths, these output LLRs are also scaled by the same factor. Which path is chosen by the algorithm as the ML path will be unaffected by this scaling of the metrics, and so the hard decisions given by the algorithm will be unaffected by using the incorrect value of L_c.

Consider now the operation of the SOVA as a component decoder within an iterative turbo decoder. Assuming that no a-priori information about the values of the bits is available to the iterative turbo decoder, the first component decoder in the first iteration takes channel values only. The a-priori information values $L(u_k)$ are set equal to zero. If the correct value of the channel reliability measure for the channel SNR used is L_c, but an incorrect value of $\hat{L}_c$ is used instead, then effectively the channel input values will have been scaled by a factor X, where:

$$X = \frac{\hat{L}_c}{L_c}. \tag{3.73}$$

The first component decoder will process these scaled channel values, and give soft-output LLRs $L(u_k|\underline{y})$. From our discussions above these soft outputs, and hence the extrinsic information $L_e(u_k)$ derived from them using Equation 3.71, will be equal to the correct soft outputs scaled by X. Next the second component decoder will take a-priori information, equal to the interleaved extrinsic information $L_e(u_k)$ from the first decoder and the channel inputs, and will use these values to calculate its soft outputs. Both the channel values $\hat{L}_c y_{kl}$ and the a-priori information $L(u_k)$ will have been scaled by X relative to their values if the correct L_c had been used, and so again all the metrics used in the SOVA will be scaled by X, and the soft outputs from this decoder will simply be the correct soft outputs scaled by the factor X. Hence we see that, because of the linearity in the SOVA, the effect of using an incorrect value of the channel reliability measure is that the output LLRs from the decoder are scaled by a constant factor. The relative importance of the two inputs to the decoder, i.e. the a-priori information and the channel information, will not change, since the LLRs for both these sources of information will be scaled by the same factor. The soft outputs from the final component decoder in the final iteration will have the same sign as those that would have been calculated using the correct value of L_c, and will merely have been scaled by X. Hence the hard outputs from an iterative turbo decoder using the SOVA are unaffected by the value of the channel reliability value L_c used, as can be seen from Figure 3.35.

The same linearity that is present in the SOVA is also found in the Max-Log-MAP algorithm. Instead of one metric two are calculated, but again only simple additions of the cross-correlation of the expected and received channel values are used. Hence, if an incorrect value of the channel reliability value is used, all the metrics are simply scaled by a factor X. As in the SOVA, the soft outputs are given by the differences in metrics between different paths, and so the same argument as was used above for the SOVA will also apply to the Max-Log-MAP algorithm:- if the input channel values are scaled by X, then the soft outputs of all the component decoders will also be scaled by X, and the final hard decisions given by the turbo decoder will be unaffected.

Let us now consider the Log-MAP algorithm. This is identical to the Max-Log-MAP algorithm, except for a correction factor $f_c(x) = \ln(1 + \mathrm{e}^{-x})$ used in the calculation of the forward and backward metrics $A_k(s)$ and $B_k(s)$ and the soft-output LLRs. The function $f_c(x)$ is non-linear — it decreases asymptotically towards zero as x increases. Hence the linearity that is present in the Max-Log-MAP and SOVA algorithms is not present in the Log-MAP algorithm. We found that the effect of this non-linearity is two-fold if an incorrect value of the channel reliability value is used. First, even when only the channel values are

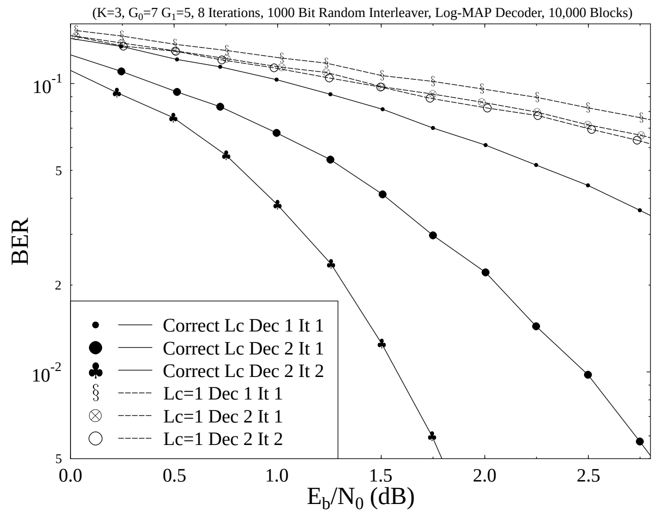


Figure 3.36: BER within an iterative turbo decoder using the Log-MAP decoder and the parameters of Table 3.5, with correct and incorrect channel reliability values L_c.

used to calculate the soft outputs from the algorithm, the component decoder makes more hard-decision errors than if the correct value of L_c were used. This is the case for the first component decoder in the first iteration, where the a-priori information values $L(u_k)$ are assumed to be equal to zero. Figure 3.36 shows the performance of an iterative turbo decoder using the Log-MAP algorithm after the first component decoder in the first iteration, denoted by 'Dec 1 It 1' in the key, when both the correct value of the channel reliability measure L_c and an incorrect value of $L_c = 1$ are used. It can be seen from this figure that when the channel reliability value $L_c = 1$ is used the BER for the first component decoder in the first iteration is significantly increased.

As well as making more hard-decision errors, if an incorrect value of the channel reliability measure is used with the Log-MAP algorithm, then the extrinsic information derived from the soft-output values from the first component decoder has incorrect amplitudes. This means that the a-priori information that is used by the second decoder in the first iteration, and by both decoders in subsequent iterations, will have incorrect amplitudes relative to the soft channel inputs. In an iterative turbo decoder the feeding of a-priori information from one component decoder to the next allows a rapid decrease in the BER of the decoder as the number of iterations increases, as was seen in Figure 3.21. However, when the incorrect value of L_c is used in an iterative turbo decoder employing the Log-MAP algorithm, owing to the incorrect scaling of the a-priori information relative to the channel inputs, no such rapid fall in the BER with the number of iterations occurs. In fact the performance of the decoder is largely unaffected by the number of iterations used. This can be seen from Figure 3.36, which shows the BER from the second decoder after one and two iterations. A significant performance improvement can be observed between the first and second iterations when the correct value of L_c is used. By contrast, when the value of $L_c = 1$ is used, there is only a marginal improvement between the first and second iteration.

In reference [160] Summers and Wilson consider the degradation in the performance of an iterative turbo decoder using the MAP algorithm when the channel SNR is not correctly estimated. As explained in Section 3.3, the MAP and Log-MAP algorithms give identical outputs and hence our analysis of the Log-MAP algorithm also applies to the MAP algorithm. In [160] the authors propose a method for blind estimation of the channel SNR, using the ratio of the average squared received channel value to the square

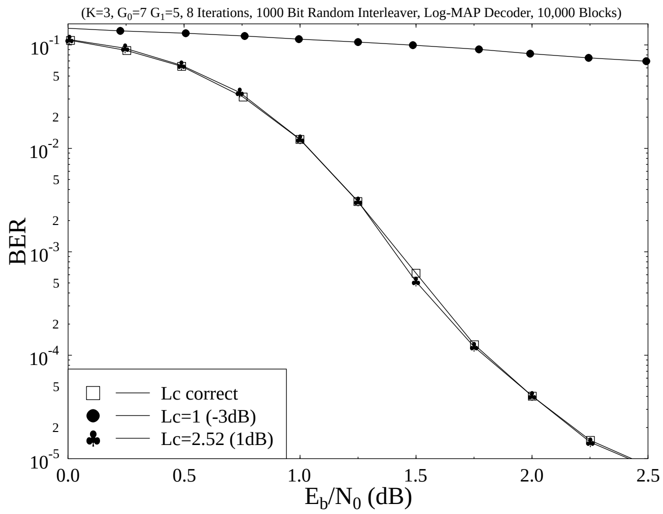


Figure 3.37: Turbo decoder performance using the Log-MAP algorithm and the parameters of Table 3.5 with a constant estimated channel reliability measure L_c.

of the average of the magnitudes of the received channel values. For a one-third-rate code and a block length of 420 data bits (so 1260 coded bits are transmitted) it is shown that the SNR derived using this method rarely differs from the true SNR by more than 3 dB. It is also shown that using these estimated SNRs to derive a channel reliability measure gives a turbo decoder performance using the MAP algorithm almost identical to that given using channel reliability measures derived from the true SNR.

Our work presented here shows that if the Max-Log-MAP or SOVA algorithms are used as the component decoders, then no such SNR estimation is necessary for a turbo decoder. If the MAP, or equivalently the Log-MAP, algorithm is used then, as can be seen from Figure 3.35, if a very inaccurate value of L_c is used the turbo decoder performance will be drastically affected. However, we have found that the value of L_c used does not have to be very close to the true value for a good BER performance to be obtained. Figure 3.37 compares the performance of a turbo decoder using the Log-MAP algorithm with the correct value of L_c, to that of a scheme which uses a constant value of $L_c = 2.52$. This corresponds to a value of E_b/N_0 of 1 dB. It can be seen that using this estimated value of L_c gives a performance virtually identical to that given with the correct value of L_c for values of E_b/N_0 from 0 to 2.5 dB, or BERs from 10^{-1} to 10^{-5}. Hence, even when using the Log-MAP algorithm, only a rough estimate of L_c is needed.

Having investigated the performance of turbo codes when used with BPSK modulation over AWGN channels, in the next section we discuss the use of both convolutional and turbo codes with higher-order modulation schemes. This allows turbo codes to be used in systems which are both bandwidth and power efficient.

3.5 Turbo Coding Performance over Rayleigh Channels

3.5.1 Introduction

In the previous sections we have discussed the performance of turbo coding in conjunction with various modulation constellations over AWGN channels. We now move on to an investigation of using turbo coding

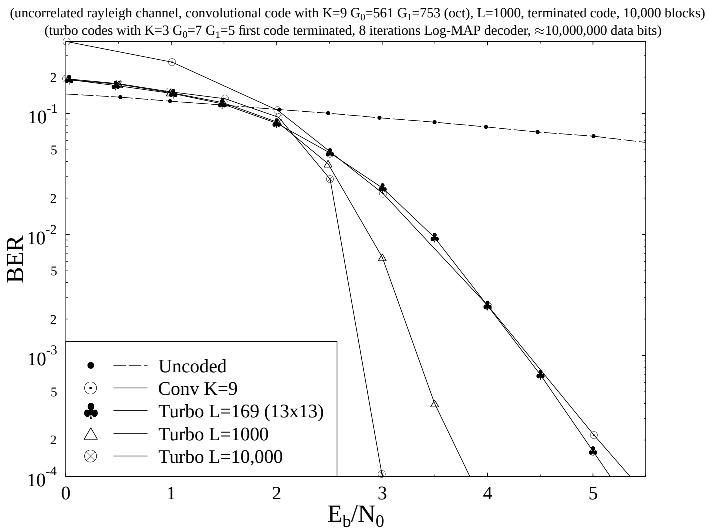


Figure 3.38: BER performance of turbo codes with different frame lengths L over perfectly interleaved Rayleigh fading channels. Other turbo codec parameters as in Table 3.5.

over fading channels. In this work we have assumed Rayleigh fading, and that the receiver has exact estimates of the fading amplitude and phase inflicted by the channel. This assumption is justified, since several techniques, such as for example Pilot Symbol Assisted Modulation (PSAM) [161], are available for providing practical mechanisms of Channel State Information (CSI) recovery, achieving a performance close to that assuming perfect CSI recovery. In Section 3.5.2 we look at the performance of various turbo codes over Rayleigh fading channels which are perfectly interleaved. Then in Section 3.5.3 we consider the effects correlations experienced in real Rayleigh fading channels have, and evaluate the performance of turbo codes in such channels.

3.5.2 Performance over Perfectly Interleaved Narrowband Rayleigh Channels

Figure 3.38 shows the performance of three turbo codes with different frame lengths L over a perfectly interleaved Rayleigh fading channel using BPSK modulation. All the turbo codecs use two $K = 3$ RSC component codes with generator polynomials $G_0 = 7$ and $G_1 = 5$. At the decoder eight iterations of the Log-MAP decoder are used. Also shown in Figure 3.38 is the performance of a constraint length $K = 9$ convolutional code which, as explained earlier, has a decoder complexity which is similar to or slightly higher than that of the turbo decoder. It can be seen that the turbo codes with frame lengths of $L = 1000$ or $L = 10000$ give a significant increase in performance over the convolutional code. Even the turbo code with a short frame length of 169 bits outperforms the convolutional code for BERs below 10^{-3}.

Comparing the performance of the $L = 169$, $L = 1000$ and $L = 10,000$ turbo codes in Figure 3.38 to those in Figure 3.26 for the same codes over an AWGN channel, we see that the perfectly interleaved fading of the received channel values degrades the BER performance of the code by around 2 dB at a BER of 10^{-4}, with a larger degradation for the shorter frame length codes.

The short frame length $L = 169$ turbo codec in Figure 3.38 uses a 13×13 block interleaver. We found in Section 3.4.6 that when communicating over a Gaussian channel, for a half-rate turbo code having short

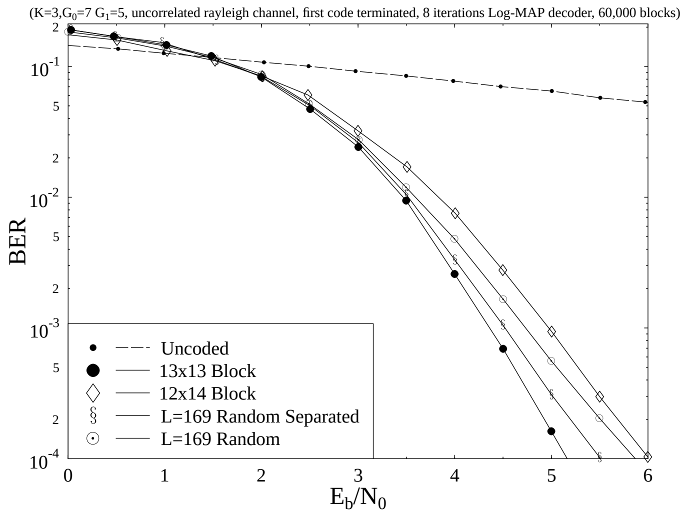


Figure 3.39: BER performance of turbo codes with different interleavers over perfectly interleaved Rayleigh fading channels. Other turbo codec parameters as in Table 3.5.

frame lengths, a block interleaver with an odd number of rows and columns should be used. Figure 3.39 shows the effect of using different interleavers, all with a frame length of approximately 169 bits, on the BER performance of a turbo codec over a perfectly interleavered Rayleigh channel. It can be seen from this figure that again for the short frame length of 169 bits the best performance is given by a block interleaver with an odd number of rows and columns. This block interleaver acheives odd–even separation [68] so that each data bit has one, and only one, of the two parity bits associated with it transmitted. The 12×14 block interleaver shown in Figure 3.39 does not give odd–even separation, and so even though its frame length is almost identical to that of the 13×13 block interleaver (168 rather than 169 bits) it performs almost 1 dB worse than the 13×13 block interleaver at a BER of 10^{-4}. The two random interleavers shown in Figure 3.39 also perform worse than the 13×13 block interleaver, although the random interleaver with odd–even separation does perform better than the non-separated interleaver.

Figure 3.40 shows how the choice of the component decoders used at the turbo decoder affects the performance of the codec over a perfectly interleavered Rayleigh channel. It can be seen that again the Log-MAP decoder gives the best performance, followed by the Max-Log-MAP decoder, with the SOVA decoder, the simplest of the three, giving the worst performance. It can also be seen that the differences in performances between the different decoders are slightly larger than they were over an AWGN channel — the Max-Log-MAP decoder performs about 0.2 dB worse than the Log-MAP decoder, and the SOVA decoder is about 0.8 dB worse than the Log-MAP decoder.

Figure 3.41 shows the effect of puncturing on a turbo code with frame length $L = 1000$ over the perfectly interleaved Rayleigh channel. In Figure 3.22 we saw that over the AWGN channel the third-rate code outperformed the half-rate code by about 0.6 dB in terms of E_b/N_0. We see from Figure 3.41 that again for the perfectly interleaved Rayleigh channel the difference in performance is bigger — about 1.5 dB in terms of E_b/N_0 or about 3.25 dB in terms of channel SNR.

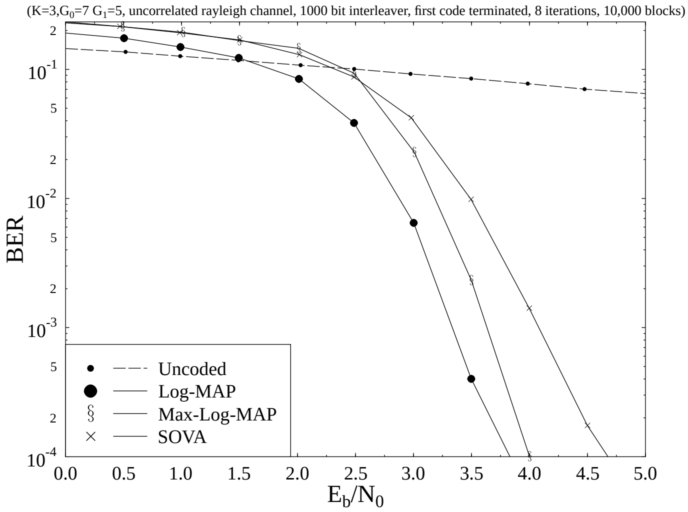


Figure 3.40: BER performance of turbo codes with $L = 1000$ using different component decoders over perfectly interleaved Rayleigh fading channels. Other turbo codec parameters as in Table 3.5.

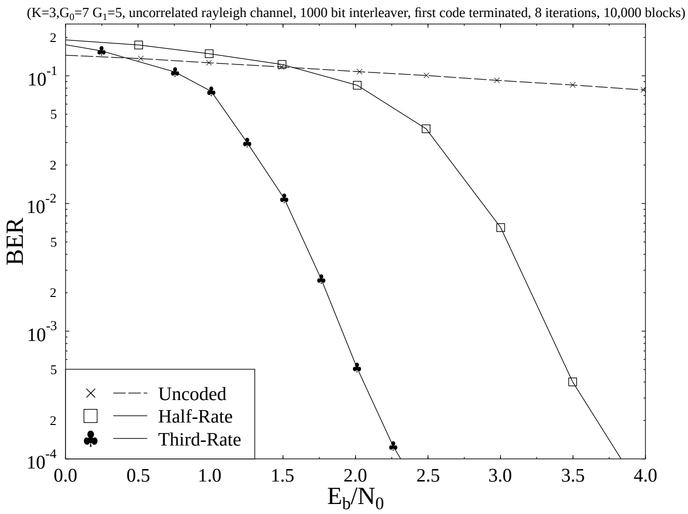


Figure 3.41: The BER performance comparison between one-third- and one-half-rate of turbo codes over perfectly interleaved Rayleigh fading channels. Other turbo codec parameters as in Table 3.5.

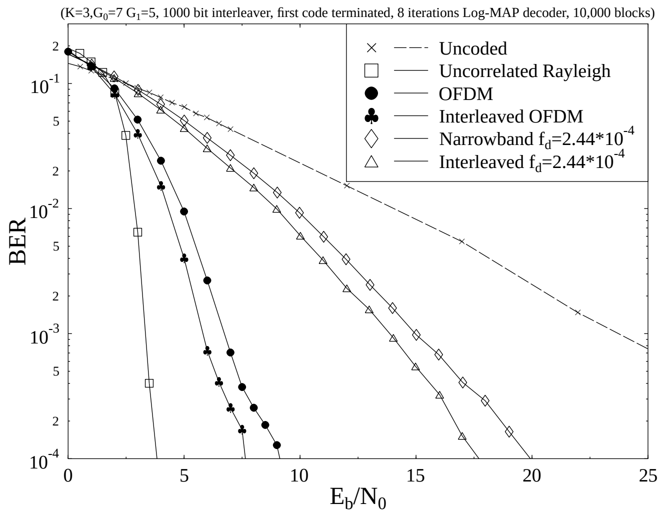


Figure 3.42: Performance of turbo coding over Rayleigh fading channels. Turbo codec parameters as in Table 3.5.

3.5.3 Performance over Correlated Narrowband Rayleigh Channels

Figure 3.42 shows the performance of a half-rate turbo coding system with $L = 1000$ over various Rayleigh fading channels. It can be seen that by far the best performance is achieved over the perfectly interleaved Rayleigh channel, where there is no correlation between successive fading values. The narrowband Rayleigh channel exhibits a normalised Doppler frequency of $f_d = 2.44 * 10^{-4}$, since we assumed a carrier frequency of 1.9 GHz, a symbol rate of 360 KBaud and a vehicular speed of 50 km/h. It can be seen that the turbo codes give a significant coding gain over the uncoded BER results even for this channel. We found that for Rayleigh fading channels exhibiting faster fading, i.e. a higher normalised Doppler frequency, the coding gain increased. Furthermore, it can be seen that interleaving the output bits of the turbo encoder before transmission over the Rayleigh fading channel improves the performance for the narrowband system by about 2.5 dB at a BER of 10^{-4}. This gain was acheived by merely interleaving over the 2000-bit length of the output block of the turbo encoder. Higher interleaving gains can be achieved at the cost of extra delay, by interleaving over longer periods. Near-perfect interleaving over a significantly longer period would give the performance indicated by the uncorrelated Rayleigh curve in Figure 3.42.

Also shown in Figure 3.42 is the performance of our turbo codec in the context of an Orthogonal Frequency Division Multiplexing (OFDM) system communicating over Rayleigh fading channels. The performance of turbo coded OFDM will be explored in more depth in various parts of the book, but suffice to say here that OFDM achieves a good turbo coded performance, which is close to that recorded, when communicating over the perfectly interleaved Rayleigh channel. Again, interleaving over the 2000 output bits of the turbo encoder substantially improves the achievable coded performance.

3.6 Summary and Conclusions

In this chapter we have characterised the performance of turbo coding schemes in conjunction with BPSK modulation, when communicating over both AWGN and Rayleigh channels. As expected, the turbo codes have been shown to perform significantly better than convolutional codes. We have demonstrated the effects of the various decoding algorithms, the constraint length and generator polynomials of the constituent codes, as well as the influence of the transmission frame length on the achievable performance. Furthermore, based on our detailed investigations we have demonstrated the importance of the choice of the interleaver in the context of turbo codes. More explicitly, we have reached the following conclusions regarding the choice of interleavers:

- When block interleavers are used in conjunction with half-rate codes, an odd number of rows and columns should be used.
- For long frame length systems random interleavers perform better than block interleavers, but for shorter frame length systems, such as those that might be used for speech transmission, block interleavers perform better.

Finally, in Section 3.5 we provided performance results obtained when using turbo codes in conjunction with BPSK and QPSK modulation for transmissions over Rayleigh fading channels.

Part II

Space-Time Block and Space-Time Trellis Coding

Part III

Turbo Equalisation

Part IV

Coded and Space-Time-Coded Adaptive Modulation: TCM, TTCM, BICM, BICM-ID and MLC

Chapter 13

Coded Modulation Theory and Performance [1]

S. X. Ng, L. Hanzo

13.1 Introduction

In this chapter our elaborations are related to a set of combined error control techniques, where channel coding and modulation are carried out jointly. At the receiver it is possible to employ both iterative and non-iterative detection and, as expected, the latter family of iterative decoders typically achieves a better performance, although at the cost of an increased implementational complexity.

Since we are relying on convolutional coding principles, familiarity with the basic concepts of Chapters 2 and 3 is assumed.

The radio spectrum is a scarce resource. Therefore, one of the most important objectives in the design of digital cellular systems is the efficient exploitation of the available spectrum, in order to accommodate the ever-increasing traffic demands. Trellis-Coded Modulation (TCM) [290], which will be detailed in Section 13.2, was proposed originally for Gaussian channels, but it was further developed for applications in mobile communications [291, 292]. Turbo Trellis-Coded Modulation (TTCM) [92], which will be augmented in Section 13.4, is a more recent joint coding and modulation scheme that has a structure similar to that of the family of power-efficient binary turbo codes [12], but employs TCM schemes as component codes. TTCM [92] requires approximately 0.5 dB lower Signal-to-Noise Ratio (SNR) at a Bit Error Ratio (BER) of 10^{-4} than binary turbo codes when communicating using 8-level Phase Shift Keying (8PSK) over Additive White Gaussian Noise (AWGN) channels. TCM and TTCM invoked Set Partitioning (SP) based signal labelling, as will be discussed in the context of Figure 13.7 in order to achieve a higher Euclidean distance between the unprotected bits of the constellation, as we will show during our further discourse. It was shown in [290] that parallel trellis transitions can be associated with the unprotected information bits; as we will augment in Figure 13.2(b), this reduced the decoding complexity. Furthermore, in our TCM and TTCM oriented investigations random symbol interleavers, rather than bit interleavers, were utilised, since these schemes operate on the basis of symbol, rather than bit, decisions.

Turbo Coding, Turbo Equalisation and Space-Time Coding
L.Hanzo, T.H. Liew, B.L. Yeap,
 ISBN 0-470-84726-3

Another coded modulation scheme distinguishing itself by utilising bit-based interleaving in conjunction with Gray signal constellation labelling is referred to as Bit-Interleaved Coded Modulation (BICM) [84]. More explicitly, BICM combines conventional convolutional codes with several independent bit interleavers, in order to increase the achievable diversity order to the binary Hamming distance of a code for transmission over fading channels [84], as will be shown in Section 13.5.1. The number of parallel bit interleavers equals the number of coded bits in a symbol for the BICM scheme proposed in [84]. The performance of BICM is better than that of TCM over uncorrelated or perfectly interleaved narrowband Rayleigh fading channels, but worse than that of TCM in Gaussian channels owing to the reduced Euclidean distance of the bit-interleaved scheme [84], as will be demonstrated in Section 13.5.1. Recently iterative joint decoding and demodulation assisted BICM (BICM-ID) was proposed in an effort to further increase the achievable performance [90, 293–297], which uses SP-based signal labelling. The approach of BICM-ID is to increase the Euclidean distance of BICM, as will be shown in Section 13.6, and hence to exploit the full advantage of bit interleaving with the aid of soft-decision feedback-based iterative decoding [298].

In this chapter, we are going to study the properties of the above-mentioned TCM, TTCM, BICM and BICM-ID schemes. Their performance will be evaluated in Section 13.7.

13.2 Trellis-Coded Modulation

The basic idea of TCM is that instead of sending a symbol formed by m information bits, for example two information bits for 4-level Phase Shift Keying (4PSK), we introduce a parity bit, while maintaining the same effective throughput of 2 bits/symbol by doubling the number of constellation points in the original constellation to eight, i.e. by extending it to 8PSK. As a consequence, the redundant bit can be absorbed by the expansion of the signal constellation, instead of accepting a 50% increase in the signalling rate, i.e. bandwidth. A positive coding gain is achieved when the detrimental effect of decreasing the Euclidean distance of the neighbouring phasors is outweighted by the coding gain of the convolutional coding incorporated.

Ungerböck has written an excellent tutorial paper [299], which fully describes TCM, and which this section is based upon. TCM schemes employ redundant non-binary modulation in combination with a finite state Forward Error Correction (FEC) encoder, which governs the selection of the coded signal sequences. Essentially the expansion of the original symbol set absorbs more bits per symbol than required by the data rate, and these extra bit(s) are used by a convolutional encoder which restricts the possible state transitions amongst consecutive phasors to certain legitimate constellations. In the receiver, the noisy signals are decoded by a trellis-based soft-decision maximum likelihood sequence decoder. This takes the incoming data stream and attempts to map it onto each of the legitimate phasor sequences allowed by the constraints imposed by the encoder. The best fitting symbol sequence having the minimum Euclidean distance from the received sequence is used as the most likely estimate of the transmitted sequence.

Simple four-state TCM schemes, where the four-state adjective refers to the number of possible states that the encoder can be in, are capable of improving the robustness of 8PSK-based TCM transmission against additive noise in terms of the required SNR by 3dB compared to conventional uncoded 4PSK modulation. With the aid of more complex TCM schemes the coding gain can reach 6 dB [299]. As opposed to traditional error correction schemes, these gains are obtained without bandwidth expansion, or without the reduction of the effective information rate. Again, this is because the FEC encoder's parity bits are absorbed by expanding the signal constellation in order to transmit a higher number of bits per symbol. The term 'trellis' is used, because these schemes can be described by a state transition diagram similar to the trellis diagrams of binary convolutional codes [300]. The difference is that in the TCM scheme the trellis branches are labelled with redundant non-binary modulation phasors, rather than with binary code symbols.

13.2.1 TCM Principle

We now illustrate the principle of TCM using the example of a four-state trellis code for 8PSK modulation, since this relatively simple case assists us in understanding the principles involved.

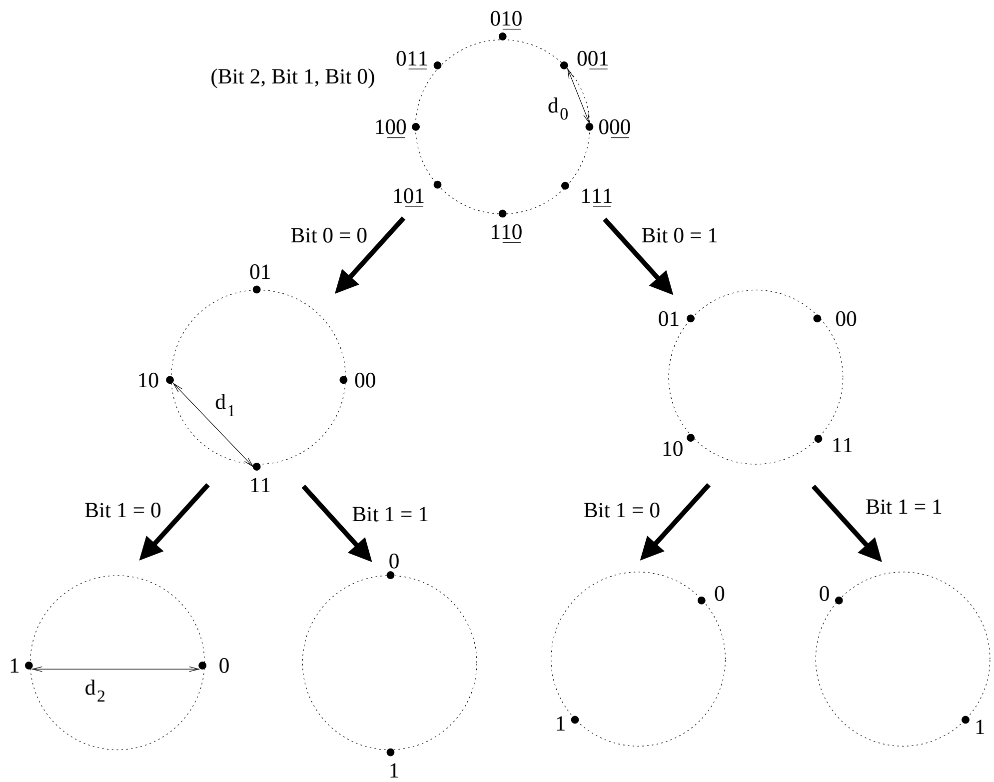


Figure 13.1: 8PSK set partitioning [290] ©IEEE, 1982, Ungerböck.

The partitioned signal set proposed by Ungerböck [290, 299] is shown in Figure 13.1, where the binary phasor identifiers are now not Gray encoded. Observe in the figure that the Euclidean distance amongst constellation points is increased at every partitioning step. The underlined last two bits, namely bit 0 and bit 1, are used for identifying one of the four partitioned sets, while bit 2 finally pinpoints a specific phasor in each partitioned set.

The signal sets and state transition diagrams for (a) uncoded 4PSK modulation and (b) coded 8PSK modulation using four trellis states are given in Figure 13.2, while the corresponding four-state encoder-based modulator structure is shown in Figure 13.3. Observe that after differential encoding bit 2 is fed directly to the 8PSK signal mapper, whilst bit 1 is half-rate convolutionally encoded by a two-stage four-state linear circuit. The convolutional encoder adds the parity bit, bit 0, to the sequence, and again these two protected bits are used for identifying which constellation subset the bits will be assigned to, whilst the more widely spaced constellation points will be selected according to the unprotected bit 2.

The trellis diagram for 4PSK is a trivial one-state trellis, which portrays uncoded 4PSK from the viewpoint of TCM. Every connected path through the trellis represents a legitimate signal sequence where no redundancy-related transition constraints apply. In both systems, starting from any state, four transitions can occur, as required for encoding two bits/symbol. The four parallel transitions in the state trellis diagram of Figure 13.2(a) do not restrict the sequence of 4PSK symbols that can be transmitted, since there is no channel coding and therefore all trellis paths are legitimate. Hence the optimum detector can only make nearest-phasor-based decisions for each individual symbol received. The smallest distance between the 4PSK phasors is $\sqrt{2}$, denoted as d_0, and this is termed the free distance of the uncoded 4PSK constellation. Each 4PSK symbol has two nearest neighbours at this distance. Each phasor is represented by a two-bit symbol and transitions from any state to any other state are legitimate.

The situation for 8PSK TCM is a little less simplistic. The trellis diagram of Figure 13.2(b) is consti-

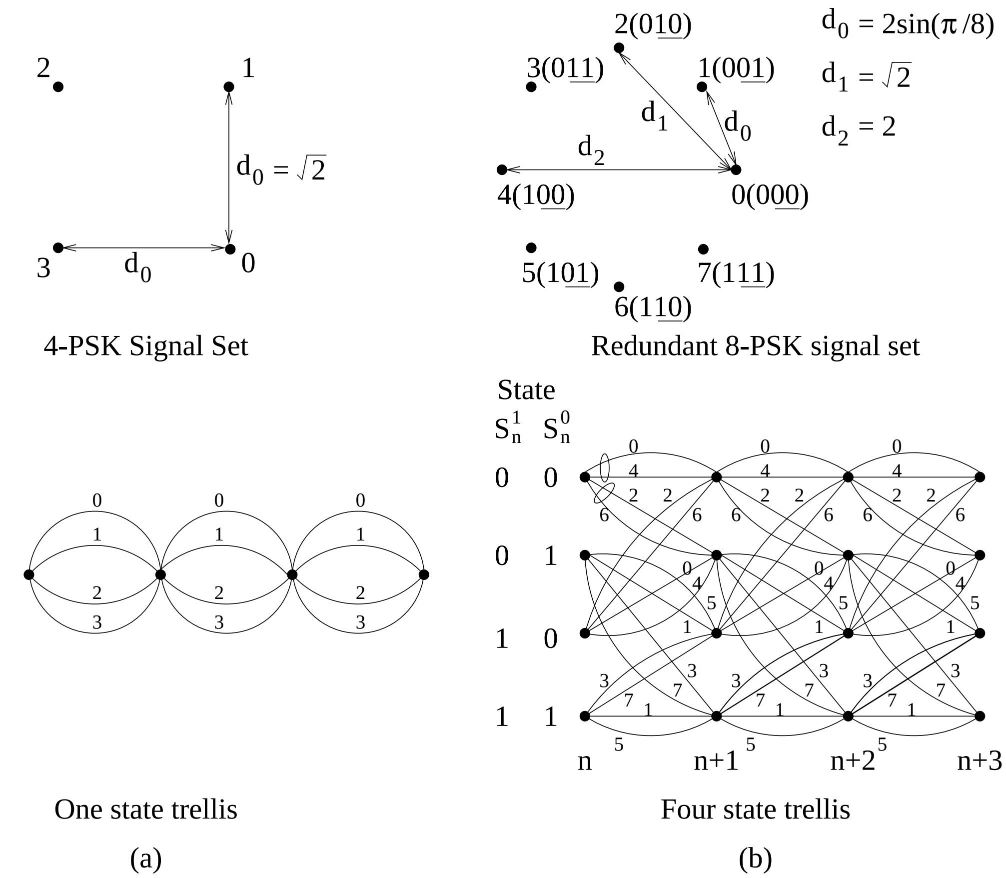


Figure 13.2: Constellation and trellis for 4- and 8PSK [299] ©IEEE, 1982, Ungerböck.

tuted by four states according to the four possible states of the shift-register encoder of Figure 13.3, which we represent by the four vertically stacked bold nodes. Following the elapse of a symbol period a new two-bit input symbol arrives and the convolutional encoder's shift register is clocked. This event is characterised by a transition in the trellis from state S_n to state S_{n+1}, tracking one of the four possible paths corresponding to the four possible input symbols.

In the four-state trellis of Figure 13.2(b) associated with the 8PSK TCM scheme, the trellis transitions occur in pairs and the states corresponding to the bold nodes are represented by the shift-register states S_n^0 and S_n^1 in Figure 13.3. Owing to the limitations imposed by the convolutional encoder of Figure 13.3 on the legitimate set of consecutive symbols only a limited set of state transitions associated with certain phasor sequence is possible. These limitations allow us to detect and to reject illegitimate symbol sequences, namely those which were not legitimately produced by the encoder, but rather produced by the error-prone channel. For example, when the shift register of Figure 13.3 is in state (0,0), only the transitions to the phasor points (0,2,4,6) are legitimate, whilst those to phasor points (1,3,5,7) are illegitimate. This is readily seen, because the linear encoder circuit of Figure 13.3 cannot produce a non-zero parity bit from the zero-valued input bits and hence the symbols (1,3,5,7) cannot be produced when the encoder is in the all-zero state. Observe in the 8PSK constellation of Figure 13.2(b) that the underlined bit 1 and bit 0 identify four twin-phasor subsets, where the phasors are opposite to each other in the constellation and hence have a high intra-subset separation. The unprotected bit 2 is then invoked for selecting the required phasor point within the subset. Since the redundant bit 0 constitutes also one of the shift-register state bits, namely S_n^0, from the

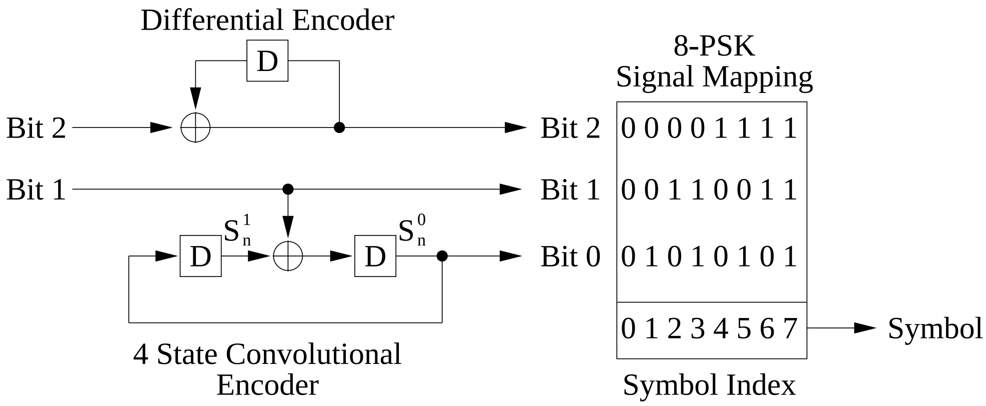


Figure 13.3: Encoder for the four-state 8PSK trellis [299] ©IEEE, 1982, Ungerböck.

initial states of (S_n^1,S_n^0) = (0,0) or (1,0) only the even-valued phasors (0,2,4,6) having $S_n^0 = 0$ can emerge, as also seen in Figure 13.2(b). Similarly, if we have (S_n^1,S_n^0) = (0,1) or (1,1) associated with $S_n^0 = 1$ then the branches emerging from these lower two states of the trellis in Figure 13.2(b) can only be associated with the odd-valued phasors of (1,3,5,7).

There are other possible codes, which would result in for example four distinct transitions from each state to all possible successor states, but the one selected here proved to be the most effective [299]. Within the 8PSK constellation we have the following distances: $d_0 = 2\sin(\pi/8)$, $d_1 = \sqrt{2}$ and $d_2 = 2$. The 8PSK signals are assigned to the transitions in the four-state trellis in accordance with the following rules:

1) Parallel trellis transitions are associated with phasors having the maximum possible distance, namely (d_2), between them, which is characteristic of phasor points in the subsets (0,4), (1,5), (2,6) and (3,7). Since these parallel transitions belong to the same subset of Figure 13.2(b) and are controlled by the unprotected bit 2, symbols associated with them should be as far apart as possible.

2) All four-state transitions originating from, or merging into, any one of the states are labelled with phasors having a distance of *at least* $d_1 = \sqrt{2}$ between them. These are the phasors belonging to subsets (0,2,4,6) or (1,3,5,7).

3) All 8PSK signals are used in the trellis diagram with equal probability.

Observe that the assignment of bits to the 8PSK constellation of Figure 13.2(b) does not obey Gray coding and hence adjacent phasors can have arbitrary Hamming distances between them. The bit mapping and encoding process employed was rather designed for exploiting the high Euclidean distances between sets of points in the constellation. The underlined bit 1 and bit 0 of Figure 13.2(b) representing the convolutional codec's output are identical for all parallel branches of the trellis. For example, the branches labelled with phasors 0 and 4 between the identical consecutive states of (0,0) and (0,0) are associated with (bit 1)=0 and (bit 0)=0, while the uncoded bit 2 can be either '0' or '1', yielding the phasors 0 and 4, respectively. However, owing to appropriate code design this unprotected bit has the maximum protection distance, namely $d_2 = 2$, requiring the corruption of phasor 0 into phasor 4, in order to inflict a single bit error in the position of bit 2.

The effect of channel errors exhibits itself at the decoder by diverging from the trellis path encountered in the encoder. Let us consider the example of Figure 13.4, where the encoder generated the phasors 0-0-0 commencing from state (0,0), but owing to channel errors the decoder's trellis path was different from this, since the phasor sequence 2-1-2 was encountered. The so-called free distance of a TCM scheme can be computed as the lower one of two distances. Namely, the Euclidean distances between the phasors labelling the parallel branches in the trellis of Figure 13.2(b) associated with the uncoded bit(s), which is $d_2 = 2$ in our example, as well as the distances between trellis paths diverging and remerging after a number of consecutive trellis transitions, as seen in Figure 13.4 in the first and last of the four consecutive (0,0) states.

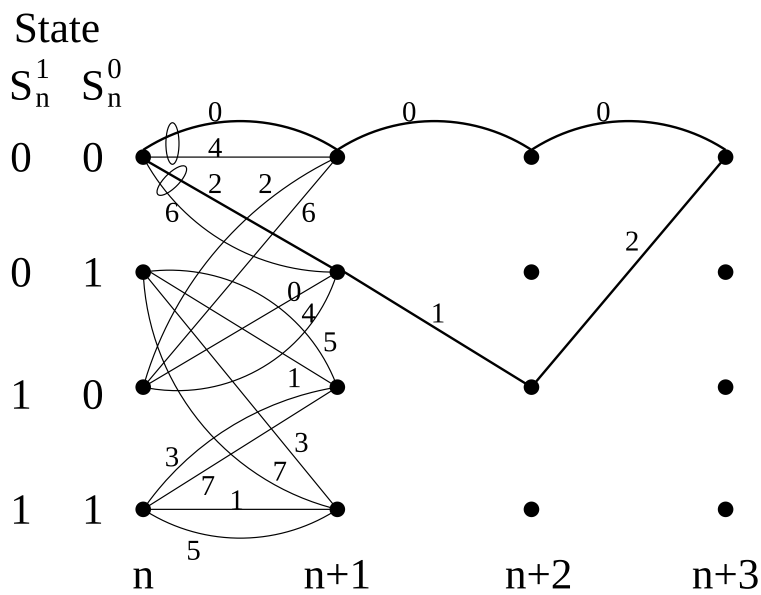


Figure 13.4: Diverging trellis paths for the computation of d_{free}. The parallel paths labelled by the symbols 0 and 4 are associated with the uncoded bits '0' and '1', respectively, as well as with the farthest phasors in the constellation of Figure 13.2(b).

The lower one of these two distances characterises the error resilience of the underlying TCM scheme, since the error event associated with it will be the one most frequently encountered owing to channel effects. Specifically, if the received phasors are at a Euclidean distance higher than half of the code's free distance from the transmitted phasor, an erroneous decision will be made. It is essential to ensure that by using an appropriate code design the number of decoded bit errors is minimised in the most likely error events, and this is akin to the philosophy of using Gray coding in a non-trellis-coded constellation.

The Euclidean distance between the phasors of Figure 13.2(b) associated with the parallel branches is $d_2 = 2$ in our example. The distance between the diverging trellis paths of Figure 13.2(b) labelled by the phasor sequences of 0-0-0 and 2-1-2 following the states {(0,0),(0,0),(0,0),(0,0)} and {(0,0),(0,1),(1,0),(0,0)} respectively, portrayed in Figure 13.4, is inferred from Figure 13.2(b) as d_1-d_0-d_1. By inspecting all the remerging paths of the trellis in Figure 13.2(b) we infer that this diverging path has the shortest accumulated Free Euclidean Distance (FED) that can be found, since all other diverging paths have higher accumulated FED from the error-free 0-0-0 path. Furthermore, this is the only path having the minimum free distance of $\sqrt{d_1^2 + d_0^2 + d_1^2}$. More specifically, the free distance of this TCM sequence is given by:

$$d_{free} = min\{d_2; \sqrt{d_1^2 + d_0^2 + d_1^2}\} \tag{13.1}$$

$$= min\{2; \sqrt{2 + (2.\sin\frac{\pi}{8})^2 + 2}\}. \tag{13.2}$$

Explicitly, since the term under the square root in Equation 13.2 is higher than $d_2 = 2$, the free distance of this TCM scheme is given ultimately by the Euclidean distance between the parallel trellis branches associated with the uncoded bit 2, i.e.:

$$d_{free} = 2. \tag{13.3}$$

The free distance of the uncoded 4PSK constellation of Figure 13.2(a) was $d_0 = \sqrt{2}$ and hence the employment of TCM has increased the minimum distance between the constellation points by a factor of

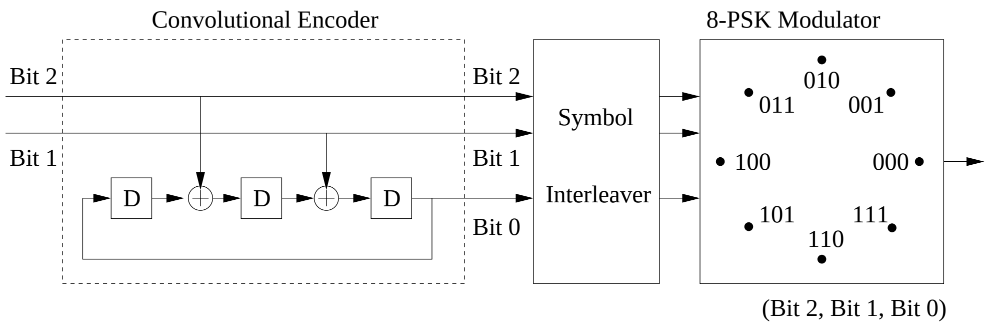


Figure 13.5: Ungerböck's RSC encoder and modulator forming the TCM encoder. The SP-based mapping of bits to the constellation points was highlighted in Figure 13.1.

$g = \frac{d_{free}^2}{d_0^2} = \frac{2^2}{(\sqrt{2})^2} = 2$, which corresponds to 3 dB. There is only one nearest-neighbour phasor at $d_{free} = 2$, corresponding to the π-rotated phasor in Figure 13.2(b). Consequently the phasor arrangement can be rotated by π, whilst retaining all of its properties, but other rotations are not admissible.

The number of erroneous decoded bits induced by the diverging path 2-1-2 is seen from the phasor constellation of Figure 13.2(b) to be 1-1-1, yielding a total of three bit errors. The more likely event of a bit 2 error, which is associated with a Euclidean distance of $d_2 = 2$, yields only a single bit error.

Soft-decision-based decoding can be accomplished in two steps. The first step is known as subset decoding, where within each phasor subset assigned to parallel transitions, i.e. to the uncoded bit(s), the phasor closest to the received channel output in terms of Euclidean distance is determined. Having resolved which of the parallel paths was more likely to have been encountered by the encoder, we can remove the parallel transitions, hence arriving at a conventional trellis. In the second step the Viterbi algorithm is used for finding the most likely signal path through the trellis with the minimum sum of squared Euclidean distances from the sequence of noisy channel outputs received. Only the signals already selected by the subset decoding are considered. For a description of the Viterbi algorithm the reader is referred to references [32, 301].

13.2.2 Optimum TCM Codes

Ungerböck's TCM encoder is a specific convolutional encoder selected from the family of Recursive Systematic Convolutional (RSC) codes [290], which attaches one parity bit to each information symbol. Only $\tilde{\mathsf{m}}$ out of m information bits are RSC encoded and hence only $2^{\tilde{\mathsf{m}}}$ branches will diverge from and merge into each trellis state. When not all information bits are RSC encoded, i.e. $\tilde{\mathsf{m}} < \mathsf{m}$, $2^{\mathsf{m}-\tilde{\mathsf{m}}}$ parallel transitions are associated with each of the $2^{\tilde{\mathsf{m}}}$ branches. Therefore a total of $2^{\tilde{\mathsf{m}}} \times 2^{\mathsf{m}-\tilde{\mathsf{m}}} = 2^{\mathsf{m}}$ transitions occur at each trellis stage. The memory length K of a code defines the number of shift-register stages in the encoder. Figure 13.5 shows the TCM encoder using an eight-state Ungerböck code [290], which has a high FED for the sake of attaining a high performance over AWGN channels. It is a systematic encoder, which attaches an extra parity bit to the original 2-bit information word. The resulting 3-bit codewords generated by the 2-bit input binary sequence are then interleaved by a symbol interleaver in order to disperse the bursty symbol errors induced by the fading channel. Then, these 3-bit codewords are modulated onto one of the $2^3 = 8$ possible constellation points of an 8PSK modulator.

The connections between the information bits and the modulo-2 adders, as shown in Figure 13.5, are given by the generator polynomials. The coefficients of these polynomials are defined as:

$$H^j(D) := h_K^j.D^K + h_{K-1}^j.D^{K-1} + \ldots + h_1^j.D + h_0^j, \tag{13.4}$$

where D represents the delay due to one register stage. The coefficient h_i^j takes the value of '1', if there is a connection at a specific encoder stage or '0', if there is no connection. The polynomial $H^0(D)$ is the feedback generator polynomial and $H^j(D)$ for $j \leq 1$ is the generator polynomial associated with the jth

Code	State	$\tilde{m}$	$H^0(D)$	$H^1(D)$	$H^2(D)$
4QAM	8	1	13	06	-
4QAM	64	1	117	26	-
8PSK	8	2	11	02	04
8PSK	32	2	45	16	34
8PSK	64	2	103	30	66
8PSK	128	2	277	54	122
8PSK	256	2	435	72	130
16QAM	64	2	101	16	64

Table 13.1: Ungerböck's TCM codes [290, 299, 302, 303].

information bit. Hence, the generator polynomial of the encoder in Figure 13.5 can be described in binary format as:

$$\begin{aligned} H^0(D) &= 1001 \\ H^1(D) &= 0010 \\ H^2(D) &= 0100, \end{aligned}$$

or equivalently in octal format as:

$$\begin{aligned} \mathbf{H}(\mathbf{D}) &= \begin{bmatrix} H^0(D) & H^1(D) & H^2(D) \end{bmatrix} \\ &= \begin{bmatrix} 11 & 02 & 04 \end{bmatrix}. \end{aligned} \tag{13.5}$$

Ungerböck suggested [290] that all feedback polynomials should have coefficients $h_K^0 = h_0^0 = 1$. This guarantees the realisability of the encoders shown in Figures 13.3 and 13.5. Furthermore, all generator polynomials should also have coefficients $h_K^j = h_0^j = 0$ for $j > 0$. This ensures that at time n the input bits of the TCM encoder have no influence on the parity bit to be generated, nor on the input of the first binary storage element in the encoder. Therefore, whenever two paths diverge from or merge into a common state in the trellis, the parity bit must be the same for these transitions, whereas the other bits differ in at least one bit [290]. Phasors associated with diverging and merging transitions therefore have at least a distance of d_1 between them, as can be seen from Figure 13.2(b). Table 13.1 summarises the generator polynomials of some TCM codes, which were obtained with the aid of an exhaustive computer search conducted by Ungerböck [299], where $\tilde{m}$ ($\leq$ m) indicates the number of information bits to be encoded, out of the m information bits in a symbol.

13.2.3 TCM Code Design for Fading Channels

It was shown in Section 13.2.1 that the design of TCM for transmission over AWGN channels is motivated by the maximisation of the FED, d_{free}. By contrast, the design of TCM concerned for transmission over fading channels is motivated by minimising the length of the shortest error event path and the product of the branch distances along that particular path [291].

The average bit error probability of TCM using M-ary PSK (MPSK) [290] for transmission over Rician channels at high SNRs is given by [291]:

$$P_b \cong \frac{1}{B} C \left(\frac{(1+\bar{K})e^{-\bar{K}}}{E_s/N_0} \right)^L ; E_s/N_0 \gg \bar{K} \tag{13.6}$$

where C is a constant that depends on the weight distribution of the code, which quantifies the number of trellises associated with all possible Hamming distances measured with respect to the all-zero path [49]. The variable B in Equation 13.6 is the number of binary input bits of the TCM encoder during each transmission interval, while $\bar{K}$ is the Rician fading parameter [49] and E_s/N_0 is the channel's symbol energy to noise

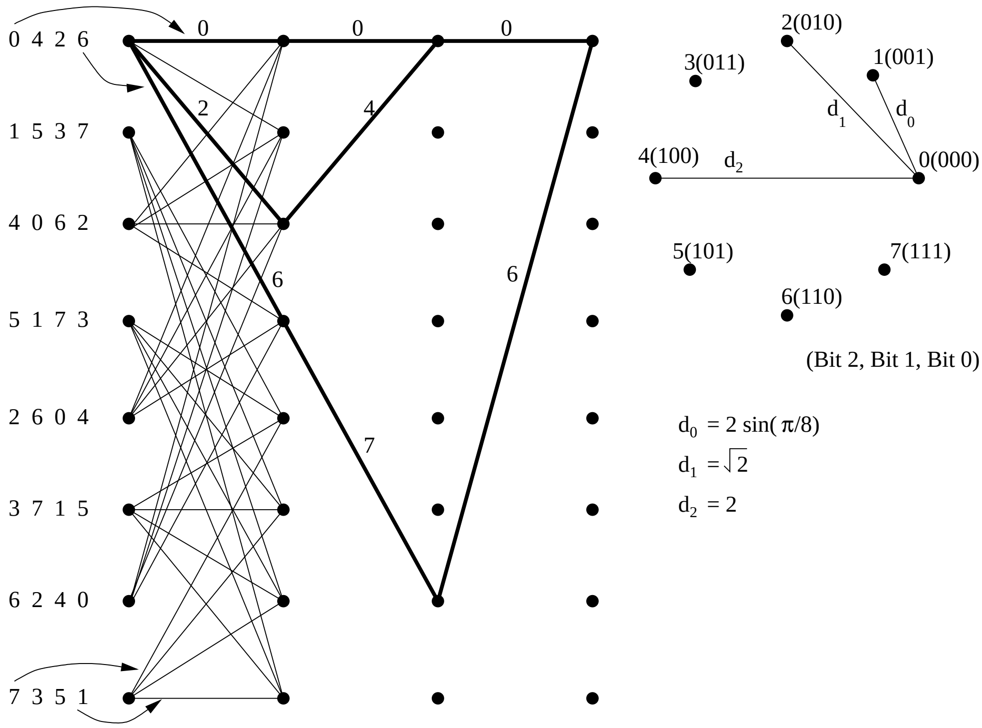


Figure 13.6: Ungerböck's eight-state 8PSK code.

spectral density ratio. Furthermore, L is the 'length' of the shortest error event path which is expressed in terms of the number of trellis stages encountered before remerging with the all-zero path. It is clear from Equation 13.6 that P_b varies inversely proportionally with $(E_s/N_0)^L$ and this ratio can be increased by increasing the code's diversity [291], which was defined in [304] as the 'length' L of the shortest error event path or the Effective Code Length (ECL). More specifically, in [304], the authors pointed out that the shortest error event paths are not necessarily associated with the minimum accumulated FED error events. For example, let the all-zero path be the correct path. Then the code characterised by the trellis seen in Figure 13.6 exhibits a minimum squared FED of:

$$\begin{aligned} d^2_{free} &= d_1^2 + d_0^2 + d_1^2 \\ &= 4.585, \end{aligned} \tag{13.7}$$

from the 0-0-0 path associated with the transmission of three consecutive 0 symbols from the path labelled with the transmitted symbols of 6-7-6. However, this is not the shortest error event path, since its length is $L = 3$, which is longer than the path labelled with transmitted symbols of 2-4, which has a length of $L = 2$ and a FED of $d^2_{free} = d_1^2 + d_0^2 + d_1^2 = 6$. Hence, the 'length' of the shortest error event path is $L = 2$ for this code, which, again, has a squared Euclidean distance of $d_1^2 + d_2^2 = 6$. In summary, the number of bit errors associated with the above $L = 3$ and $L = 2$ shortest error event paths is seven and two, respectively, clearly favouring the $L = 2$ path, which had a higher accumulated FED of 6 than that of the 4.585 FED of the $L = 3$ path. Hence, it is worth noting that if the code was designed based on the minimum FED, it may not minimise the number of bit errors. Hence, as an alternative design approach, in Section 13.5 we will study BICM, which relies on the shortest error event path L or the bit-based Hamming distance of the code and hence minimises the BER.

The design of coded modulation schemes is affected by a variety of factors. A high squared FED is desired for AWGN channels, while a high ECL and a high minimum product distance are desired for fading

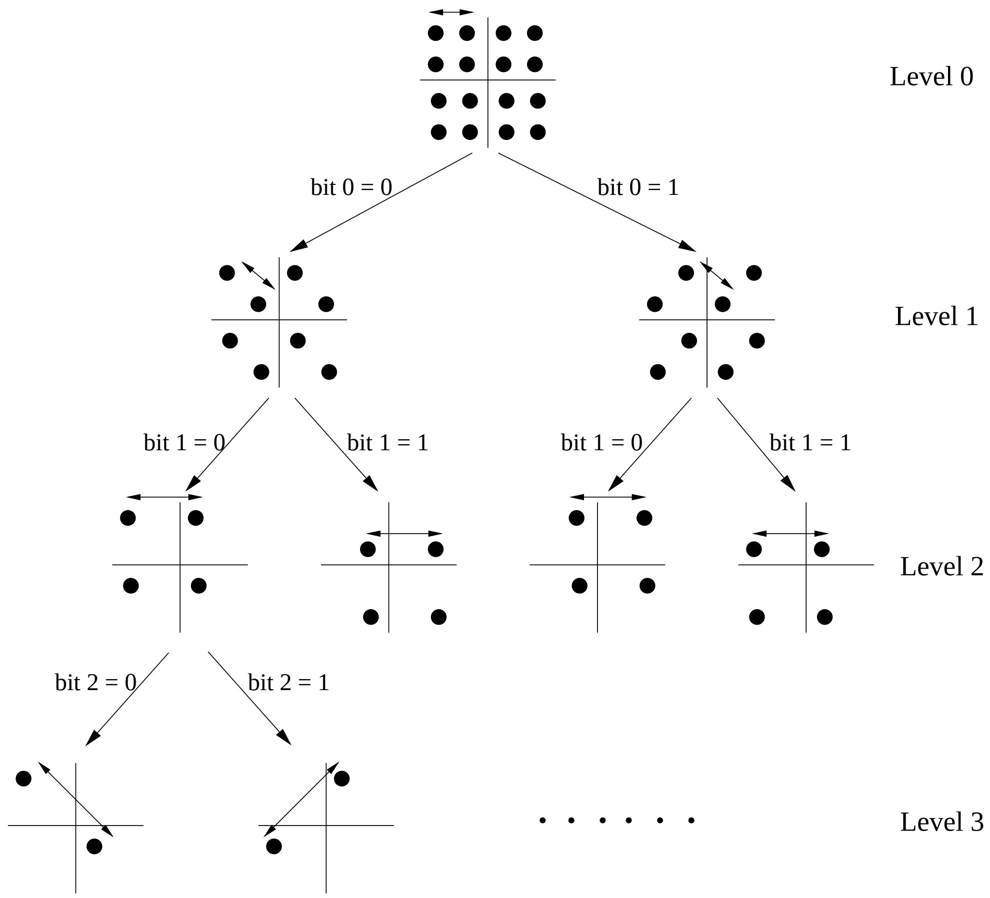


Figure 13.7: Set partitioning of a 16QAM signal constellation. The minimum Euclidean distance at a partition level is denoted by the line between the signal points [290] ©IEEE, 1982, Ungerböck.

channels [291]. In general, a code's diversity or ECL is quantified in terms of the shortest error event path L, which may be increased for example by simple repetition coding, although at the cost of reducing the effective data rate proportionately. Alternatively, space-time-coded multiple transmitter/receiver structures can be used, which increase the scheme's cost and complexity. Finally, simple interleaving can be invoked, which induces latency. In our approach, symbol-based interleaving is employed in order to increase the code's diversity.

13.2.4 Set Partitioning

As we have seen in Figure 13.4, if higher-order modulation schemes, such as 16-level Quadrature Amplitude Modulation (16QAM) or 64QAM, are used, parallel transitions may appear in the trellis diagram of the TCM scheme, when not all information bits are convolutional channel encoded or when the number of states in the convolutional encoder has to be kept low for complexity reasons. As noted before, in order to avoid encountering high error probabilities, the parallel transitions should be assigned to constellation points exhibiting a high Euclidean distance. Ungerböck solved this problem by introducing the set partitioning technique. Specifically, the signal set is split into a number of subsets, such that the minimum Euclidean distance of the signal points in the new subset is increased at every partitioning step.

In order to elaborate a little further, Figure 13.7 illustrates the set partitioning of 16QAM. Here we used the $R = \frac{3}{4}$-rate code of Table 13.1. This is a relatively high-rate code, which would not be sufficiently powerful if we employed it for protecting all three original information bits. Moreover, if we protect for example two out of the three information bits, we can use a more potent $\frac{2}{3}$-rate code for the protection of the more vulnerable two information bits and leave the most error-resilient bit of the 4-bit constellation unprotected. This is justifiable, since we can observe in Figure 13.7 that the minimum Euclidean distance of the constellation points increases from Level 0 to Level 3 of the constellation partitioning tree. This indicates that the bits labelling or identifying the specific partitions have to be protected by the RSC code, since they label phasors that have a low Euclidean distance. By contrast, the intra-set distance at Level 3 is the highest, suggesting a low probability of corruption. Hence the corresponding bit, bit 3, can be left unprotected. The partitioning in Figure 13.7 can be continued, until there is only one phasor or constellation point left in each subset. The intra-subset distance increases as we traverse down the partition tree. The first partition level, $Level$ 0, is labelled by the parity bit, and the next two levels by the coded bits. Finally, the uncoded bit labels the lowest level, $Level$ 3, in the constellation, which has the largest minimum Euclidean distance.

Conventional TCM schemes are typically decoded/demodulated with the aid of the appropriately modified Viterbi Algorithm (VA) [305]. Furthermore, the VA is a maximum likelihood sequence estimation algorithm, which does not guarantee that the Symbol Error Ratio (SER) is minimised, although it achieves a performance near the minimum SER. By contrast, the symbol-based MAP algorithm [92] guarantees the minimum SER, albeit at the cost of a significantly increased complexity. Hence the symbol-based MAP algorithm has been used for the decoding of TCM sequences. We will, however, in Section 13.4, also consider Turbo TCM (TTCM), where instead of the VA-based sequence estimation, symbol-by-symbol-based soft information has to be exchanged between the TCM decoders of the TTCM scheme. Hence in the next section we will present the symbol-based MAP algorithm.

13.3 The Symbol-based MAP Algorithm

In this section, the non-binary or symbol-based MAP decoding algorithm will be presented. The non-binary MAP algorithm was proposed in [92], while the binary MAP algorithm was first presented in [11] and it has been described in detail in Section 3.3.3. In our forthcoming discourse we use $P(e)$ to denote the probability of the event e, and, given a received symbol sequence $\underline{y}$ of length N, the received channel output symbol y_k associated with the present transition, $\underline{y}_{j<k}$, is constituted by the symbol sequence received prior to the present transition, as seen in Figure 13.9 below. Similarly, the symbol sequence $\underline{y}_{j>k}$ received after the present transition is shown in Figure 13.9. We note here in closing that while in Section 3.3.3 $\underline{y}_k$ associated with the present transition was a codeword of length n, in the context of the symbol-based MAP algorithm of this section y_k denotes a channel output sample representing a specific received symbol.

13.3.1 Problem Description

The problem that the MAP algorithm has to solve is presented in Figure 13.8. An information source produces a sequence of N information symbols u_k, $k = 1, 2, \ldots, N$. Each information symbol can assume M different values, i.e. $u_k \in \{0, 1, \ldots, M-1\}$, where M is typically a power of two, so that each information symbol carries $\mathsf{m} = \log_2 M$ information bits. We assume here that the symbols are to be transmitted over an AWGN channel. To this end, the m-bit symbols are first fed into an encoder for generating a sequence of N channel symbols $x_k \in X$, where X denotes the set of complex values belonging to some phasor constellations such as an increased-order QAM or PSK constellation, having $\bar{M}$ possible values carrying $\bar{\mathsf{m}} = \log_2 \bar{M}$ bits. Again, the channel symbols are transmitted over an AWGN channel and the received symbols are:

$$y_k = x_k + n_k, \tag{13.8}$$

where n_k represents the complex AWGN samples. The received symbols are fed to the decoder, which has the task of producing an estimate $\hat{u}_k$ of the 2^{m}-ary information sequence, based on the $2^{\bar{\mathsf{m}}}$-ary received sequence, where $\bar{\mathsf{m}} > \mathsf{m}$. If the goal of the decoder is that of minimising the number of symbol errors, where a symbol error occurs when $u_k \neq \hat{u}_k$, then the best decoder is the MAP decoder [11]. This decoder computes the A-Posteriori Probability (APP) $A_{k,m}$ for every 2^{m}-ary information symbol u_k that the information symbol value was m given the received sequence, i.e. computes $A_{k,m} = p(u_k = m|\underline{y})$, for $m = 0, 1, \ldots, M-1$, $k = 1, 2, \ldots, N$. Then it decides that the information symbol was the one having the highest probability, i.e. $\hat{u}_k = m$ if $A_{k,m} \geq A_{k,i}$ for $i = 0 \ldots M-1$. In order to realise a MAP decoder one has to devise a suitable algorithm for computing the APP.

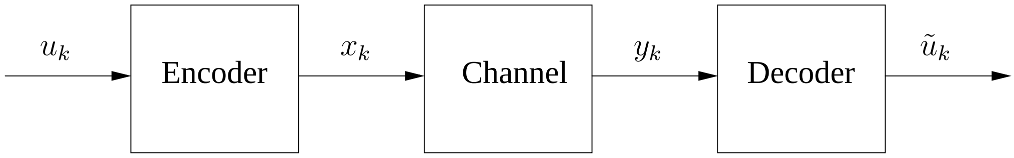


Figure 13.8: The transmission system.

In order to compute the APP, we must specify how the encoder operates. We consider a trellis encoder. The operation of a trellis encoder can be described by its trellis. The trellis seen in Figure 13.9 is constituted by $(N+1) \cdot S$ nodes arranged in $(N+1)$ columns of S nodes. There are M branches emerging from each node, which arrive at nodes in the immediately following column. The trellis structure repeats itself identically between each pair of columns.

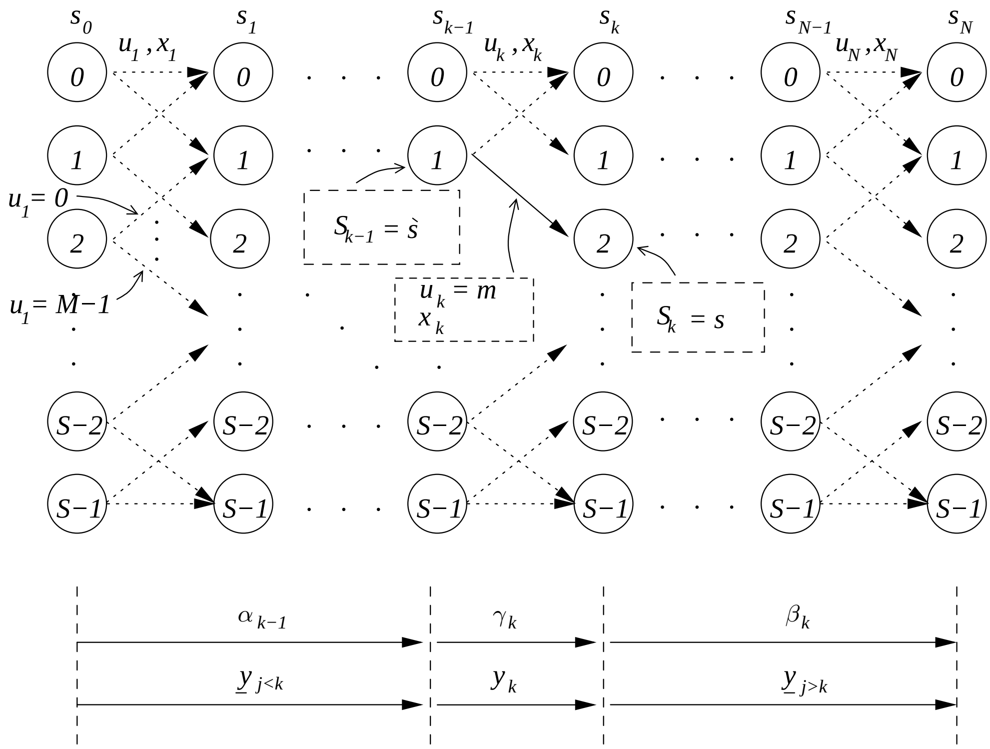


Figure 13.9: The non-binary trellis and its labelling, where there are M branches emerging from each node. (The binary trellis for the binary MAP algorithms is illustrated in Figure 3.6, where there are only two possible branches emerging from each node.)

It is possible to identify a set of paths originating from the nodes in the first column and terminating in a node of the last column. Each path will comprise exactly N branches. When employing a trellis encoder, the input sequence unambiguously determines a single path in the trellis. This path is identified by labelling the M branches emerging from each node by the M possible values of the original information symbols, although only the labelling of the first branch at $m = 0$ and the last branch at $m = M - 1$ IS shown in Figure 13.9 owing to space limitations. Then, commencing from a specified node in the first column, we use the first input symbol, u_1, to decide which branch is to be chosen. If $u_1 = m$, we choose the branch labelled with m, and move to the corresponding node in the second column that this branch leads to. In this node we use the second information symbol, u_2, for selecting a further branch and so on. In this way the information sequence identifies a path in the trellis. In order to complete the encoding operation, we have to produce the symbols to be transmitted over the channel, namely $x_1, x_2, \ldots, x_N$ from the information symbols $u_1, u_2, \ldots, u_N$. To this end we add a second label to each branch, which is the corresponding phasor constellation point that is transmitted when the branch is encountered.

In a trellis it is convenient to attach a time index to each column, from 0 to N, and to number the nodes in each column from 0 to $S - 1$. This allows us to introduce the concept of trellis states at time k. Specifically, during the encoding process, we say that the trellis is in state s at time k, and write $S_k = s$, if the path determined by the information sequence crosses the sth node of the kth column. There is a branch leading from state $S_{k-1} = \grave{s}$ to state $S_k = s$, which is encountered if the input symbol is $u_k = m$, and the corresponding transmitted symbol is x_k. The aim of the MAP decoding algorithm is to find the path in the trellis that is associated with the most likely transmitted symbols, i.e. that of minimising the SER. By contrast, the VA-based detection of TCM signals aims to identify the most likely transmitted symbol sequence, which does not automatically guarantee attaining the minimum SER.

13.3.2 Detailed Description of the Symbol-based MAP Algorithm

Having described the problem to be solved by the MAP decoder and the encoder structure, we now seek an algorithm capable of computing the APP, i.e. $A_{k,m} = P(u_k = m|\underline{y})$. The easiest way of computing these probabilities is by determining the sum of a different set of probabilities, namely $P(u_k = m \wedge S_{k-1} = \grave{s} \wedge S_k = s|\underline{y})$, where, again, $\underline{y}$ denotes the received symbol sequence. This is because we can devise a recursive way of computing the second set of probabilities as we traverse through the trellis from state to state, which reduces the detection complexity. Thus we write:

$$A_{k,m} = P(u_k = m|\underline{y}) = \sum_{\text{all } \grave{s},s} P(u_k = m \wedge S_{k-1} = \grave{s} \wedge S_k = s|\underline{y}), \tag{13.9}$$

where the summation implies adding all probabilities associated with the nodes $\grave{s}$ and s labelled by $u_k = m$ and the problem is now that of computing $P(u_k = m \wedge S_{k-1} = \grave{s} \wedge S_k = s|\underline{y})$. As a preliminary consideration we note that this probability is zero, if the specific branch of the trellis emerging from state $\grave{s}$ and merging into state s is not labelled with the input symbol m. Hence, we can eliminate the corresponding terms of the summation. Thus, we can rewrite Equation 13.9 as:

$$A_{k,m} = \sum_{\substack{(\grave{s},s)\Rightarrow \\ u_k=m}} P(S_{k-1} = \grave{s} \wedge S_k = s|\underline{y}), \tag{13.10}$$

where $(\grave{s}, s) \Rightarrow u_k = m$ indicates the specific set of transitions emerging from the previous state $S_{k-1} = \grave{s}$ to the present state $S_k = s$ that can be encountered when the input symbol is $u_k = m$. If the transitions $(\grave{s}, s) \Rightarrow u_k = m$ exist, then we can compute the probabilities $P(S_{k-1} = \grave{s} \wedge S_k = s|\underline{y})$, using Bayes' rule, as:

$$P(\grave{s} \wedge s|\underline{y}) = \frac{1}{P(\underline{y})} \cdot P(\grave{s} \wedge s \wedge \underline{y}). \tag{13.11}$$

Using Equations 3.18 to 3.20 of Section 3.3.3.1, we can rewrite Equation 13.11 as:

$$P(\grave{s} \wedge s|\underline{y}) = \frac{1}{P(\underline{y})} \cdot \beta_k(s) \cdot \gamma_k(\grave{s}, s) \cdot \alpha_{k-1}(\grave{s}), \tag{13.12}$$

where:

$$\begin{aligned}\alpha_{k-1}(\grave{s}) &= P(S_{k-1}=\grave{s}\wedge \underline{y}_{j<k})\\ \beta_k(s) &= P(\underline{y}_{j>k}|S_k=s)\\ \gamma_k(\grave{s},s) &= P(\{y_k\wedge S_k=s\}|S_{k-1}=\grave{s})\end{aligned} \tag{13.13}$$

similar to those in Equations 3.21, 3.22 and 3.23 for the binary MAP algorithm, except for the slight differences that will be discussed during our discourse below. More specifically, in Sections 3.3.3.2 and 3.3.3.3 we have shown how the $\alpha_{k-1}(\grave{s})$ values and the $\beta_k(s)$ values can be efficiently computed using the $\gamma_k(\grave{s},s)$ values. However, the computation of the $\gamma_k(\grave{s},s)$ values in the symbol-based MAP is different from that of the binary MAP. In our forthcoming discourse we study the $\gamma_k(\grave{s},s)$ values and further simplify Equation 13.10.

Upon substituting Equation 13.12 into Equation 13.10 we have:

$$A_{k,m} = C_k^1 \cdot \sum_{\substack{(\grave{s},s)\Rightarrow\\ u_k=m}} \beta_k(s)\cdot\gamma_k(\grave{s},s)\cdot\alpha_{k-1}(\grave{s}), \tag{13.14}$$

where $C_k^1 = \frac{1}{p(\underline{y})}$ is a common normalisation factor. The first simplification is to note that we do not necessarily need the exact $A_{k,m}$ values, only their ratios. In fact, for a fixed time instant k, the vector $A_{k,m}$, which is a vector of probabilities, has to sum to unity. Thus, by normalising the sum in Equation 13.10 to unity, we can compute the exact value of $A_{k,m}$ from $\bar{A}_{k,m}$ with the aid of:

$$A_{k,m} = C_k \cdot \bar{A}_{k,m}. \tag{13.15}$$

For this reason we can omit the common normalisation factor of C_k^1 in Equation 13.14.

Let us now consider the term $\gamma_k(\grave{s},s)$ of Equation 13.14, which can be rewritten using Bayes' rule as:

$$\begin{aligned}\gamma_k(\grave{s},s) &= P(\{y_k\wedge s\}|\grave{s})\\ &= P(y_k|\{\grave{s}\wedge s\})\cdot P(s|\grave{s})\\ &= P(y_k|\{\grave{s}\wedge s\})\cdot P(m),\end{aligned} \tag{13.16}$$

where $u_k = m$ is the input symbol necessary to cause the transition from state $S_{k-1}=\grave{s}$ to state $S_k = s$, and $P(m)$ is the a-priori probability of this symbol. Let us now study the multiplicative terms at the right of Equation 13.16, where $P(y_k|\{\grave{s}\wedge s\})$ is the probability that we receive y_k when the branch emerges from state $S_{k-1}=\grave{s}$ of Figure 13.9 to state $S_k = s$. When this branch is encountered, the symbol transmitted is x_k, as seen in Figure 13.9. Thus, the probability of receiving the sample y_k, given that the previous state was $S_{k-1}=\grave{s}$ and the current state is $S_k=s$, can be written as:

$$P(y_k|\{\grave{s}\wedge s\}) = p(y_k|x_k). \tag{13.17}$$

By remembering that $y_k = x_k + n_k$, where n_k is the complex AWGN, we can compute Equation 13.17 as [306]:

$$\begin{aligned}P(y_k|\{\grave{s}\wedge s\}) &= \frac{1}{2\pi\sigma^2}\cdot e^{-\frac{|y_k-x_k|^2}{2\sigma^2}}\\ &= C_k^2\cdot\eta_k(\grave{s},s),\end{aligned} \tag{13.18}$$

where $\sigma^2 = N_0/2$ is the noise's variance, N_0 is the noise's Power Spectral Density (PSD), $C_k^2 = \frac{1}{2\pi\sigma^2}$ and $\eta_k(\grave{s},s) = e^{\frac{-|y_k-x_k|^2}{2\sigma^2}}$. In verbal terms, Equation 13.18 indicates that the probability expressed in Equation 13.17 is a function of the distance between the received noisy sample y_k and the transmitted noiseless sample x_k. Observe in Equation 13.18 that we can drop the multiplicative factor of $C_k^2 = \frac{1}{2\pi\sigma^2}$, since it constitutes another scaling factor. As to the second multiplicative term on the right-hand side of

Equation 13.16, note that $p(u_k = m|S_{k-1} = \grave{s}) = p(u_k = m)$, since the original information to be transmitted is independent of the previous trellis state. The probabilities:

$$\Pi_{k,m} = p(u_k = m) \tag{13.19}$$

are the a-priori probabilities of the information symbols. Typically the information symbols are independent and equiprobable, hence $\Pi_{k,m} = 1/M$. However, if we have some prior knowledge about the transmitted symbols, this can be used as their a-priori probability. As we will see, a turbo decoder will have some a-priori knowledge about the transmitted symbols after the first iteration. We now rewrite Equation 13.16 using Equations 13.18 and 13.19 as:

$$\gamma_k(\grave{s}, s) = C_k^2 \cdot \Pi_{k,m} \cdot \eta_k(\grave{s}, s).\eta_k(j, m). \tag{13.20}$$

Then, by substituting Equation 13.20 into Equation 13.14 and and exchanging the order of summations we can portray the APPs in their final form, yielding:

$$A_{k,m} = C_k \cdot \Pi_{k,m} \cdot \sum_{\substack{(\grave{s},s)\Rightarrow \\ u_k=m}} \beta_k(s) \cdot \alpha_{k-1}(\grave{s}) \cdot \eta_k(\grave{s}, s), \tag{13.21}$$

where $C_k = C_k^1 \cdot C_k^2$ is a common scaling factor at instant k that can be dropped for the sake of simplicity without any ambiguity. Therefore, we only have to consider $\bar{A}_{k,m}$ of Equation 13.15, yielding:

$$\bar{A}_{k,m} = \Pi_{k,m} \cdot \sum_{\substack{(\grave{s},s)\Rightarrow \\ u_k=m}} \beta_k(s) \cdot \alpha_{k-1}(\grave{s}) \cdot \eta_k(\grave{s}, s). \tag{13.22}$$

The symbol-based MAP algorithm can be evaluated also in the logarithmic domain (log-domain) for the sake of reducing the computational complexity and for mitigating the numerical stability problems associated with the MAP algorithm [11], when processing small numbers representing the associated probabilities. The modifications of the symbol-based MAP algorithm required for transforming it to the log-domain are similar to that of the binary MAP algorithm, which was discussed in Section 3.3.5.

13.3.3 Symbol-based MAP Algorithm Summary

Let us now summarise the operations of the symbol-based MAP algorithm using Figure 13.10. We assume that the a-priori probabilities $\Pi_{k,m}$ in Equation 13.19 were known. These are either all equal to $1/M$ or constituted by additional external information. The first step is to compute the set of probabilities $\eta_k(\grave{s}, s)$ from Equation 13.18 as:

$$\eta_k(\grave{s}, s) = e^{-\frac{|y_k - x_k|^2}{2\sigma^2}}. \tag{13.23}$$

From these and the a-priori probabilities, the $\gamma_k(\grave{s}, s)$ values are computed according to Equation 13.20 as:

$$\gamma_k(\grave{s}, s) = \Pi_{k,m} \cdot \eta_k(\grave{s}, s). \tag{13.24}$$

The above values are then used to recursively compute the values $\alpha_{k-1}(\grave{s})$ employing Equation 3.25 as:

$$\alpha_k(s) = \sum_{\text{all } \grave{s}} \gamma_k(\grave{s}, s) \cdot \alpha_{k-1}(\grave{s}), \tag{13.25}$$

and the values $\beta_k(s)$ using Equation 3.28 as:

$$\beta_{k-1}(\grave{s}) = \sum_{\text{all } s} \beta_k(s) \cdot \gamma_k(\grave{s}, s). \tag{13.26}$$

Finally, the APP can be obtained using Equation 13.22:

$$\bar{A}_{k,m} = \Pi_{k,m} \cdot \sum_{\substack{(\grave{s},s)\Rightarrow \\ u_k=m}} \beta_k(s) \cdot \alpha_{k-1}(\grave{s}) \cdot \eta_k(\grave{s}, s). \tag{13.27}$$

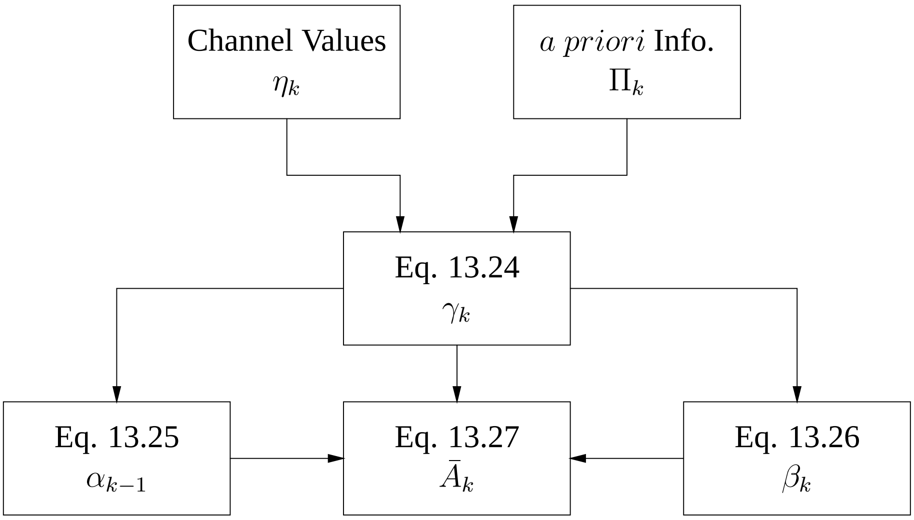


Figure 13.10: Summary of the symbol-based MAP algorithm operations. (The summary of the LLR-based binary MAP algorithm operations is shown in Figure 3.8.)

When considering the implementation of the MAP algorithm, one can opt for computing and storing the $\eta_k(\grave{s}, s)$ values, and use these values together with the a-priori probabilities for determining the values $\gamma_k(\grave{s}, s)$ during decoding. In order to compute the probabilities $\eta_k(\grave{s}, s)$ it is convenient to separately evaluate the exponential function of Equation 13.23 for every k and for every possible value of the transmitted symbol. As described in Section 13.3.1, a sequence of N information symbols was produced by the information source and each information symbol can assume M possible values, while the number of encoder states is S. There are $\bar{M} = 2 \cdot M$ possible transmitted symbols, since the size of the original signal constellation was doubled by the trellis encoder. Thus $N \cdot 2 \cdot M$ evaluations of the exponential function of Equation 13.23 are needed. Using the online computation of the $\gamma_k(\grave{s}, s)$ values, two multiplications are required for computing one additive term in each of Equations 13.25 and 13.26, and there are $N \cdot S$ terms to be computed, each requiring M terms to be summed. Hence $2 \cdot N \cdot M \cdot S$ multiplications and $N \cdot M \cdot S$ additions are required for computing the forward recursion α or the backward recursion β. Approximately three multiplications are required for computing each additive term in Equation 13.27, and there are $N \cdot M$ terms to be computed, each requiring S terms to be summed. Hence, the total implementational complexity entails $7 \cdot N \cdot M \cdot S$ multiplications, $3 \cdot N \cdot M \cdot S$ summations and $N \cdot 2 \cdot M$ exponential function evaluations, which is directly proportional to the length N of the transmitted sequence, to the number of code states S and to the number of different values M assumed by the input symbols.

The computational complexity can be reduced by implementing the algorithm in the log-domain, where the evaluation of the exponential function in Equation 13.23 is avoided. The multiplications and additions in Equations 13.24 to 13.27 are replaced by additions and Jacobian comparisons, respectively. Hence the total implementational complexity imposed is $7 \cdot N \cdot M \cdot S$ additions and $3 \cdot N \cdot M \cdot S$ Jacobian comparisons.

When implementing the MAP decoder presented here it is necessary to control the dynamic range of the likelihood terms computed in Equations 13.25 to 13.27. This is because these values tend to become lower and lower owing to the multiplication of small values. The dynamic range can be controlled by normalising the sum of the $\alpha_k(s)$ and the $\beta_k(s)$ values to unity at every particular k symbol. The resulting symbol values will not be affected, since the normalisation only affects the scaling factors C_k in Equation 13.15. However, this problem can be avoided when the MAP algorithm is implemented in the log-domain.

To conclude, let us note that the MAP decoder presented here is suitable for the decoding of finite-length, preferably short, sequences. When long sequences are transmitted, the employment of this decoder is impractical, since the associated memory requirements increase linearly with the sequence length. In this case the MAP decoder has to be modified. A MAP decoder designed for long sequences was first presented

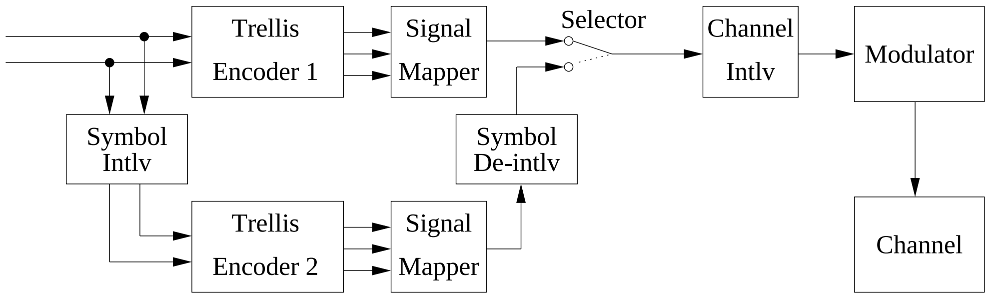


Figure 13.11: Schematic of the TTCM encoder. The selector enables the transmission of the information bits only once and selects alternative parity bits from the constituent encoders seen at the top and bottom [92] ©IEEE, 1998, Robertson and Wörz.

in [307]. An efficient implementation, derived by adapting the algorithm of [11], was proposed by Piazzo in [308]. Having described the symbol-based MAP algorithm, let us now consider Turbo TCM (TTCM) and the way it invokes the MAP procedure.

13.4 Turbo Trellis-coded Modulation

13.4.1 TTCM Encoder

It is worth describing the signal set dimensionality ($\bar{D}$) [309, 310] before we proceed. For a specific $2\bar{D}$ code, we have one $2\bar{D}$ symbol per codeword. For a general multidimensional code having a dimensionality of $\mathsf{D} = 2 \cdot n$ where $n > 0$ is an integer, one $\mathsf{D}\bar{D}$ codeword is comprised of n $2\bar{D}$ sub-codewords. The basic concept of the multidimensional signal mapping [309] is to assign more than one $2\bar{D}$ symbol to one codeword, in order to increase the spectral efficiency, which is defined as the number of information bits transmitted per channel symbol. For instance, a $2\bar{D}$ 8PSK TCM code seen in Table 13.2 maps $n = \frac{\mathsf{D}}{2} = 1$ three-bit $2\bar{D}$ symbol to one $2\bar{D}$ codeword, where the number of information bits per $2\bar{D}$ codeword is $\mathsf{m} = 2$ yielding a spectral efficiency of $\mathsf{m}/n = 2$ information bits per symbol. However, a $4\bar{D}$ 8PSK TCM code seen in Table 13.2 maps $n = \frac{\mathsf{D}}{2} = 2$ three-bit $2\bar{D}$ symbols to one six-bit $4\bar{D}$ codeword using the mapping rule of [309], where the number of information bits per $4\bar{D}$ codeword is $\mathsf{m} = 5$, yielding a spectral efficiency of $\mathsf{m}/n = 2.5$ information bits per symbol. However, during our further discourse we only consider $2\bar{D}$ signal sets.

Employing TTCM [92] avoids the obvious disadvantage of rate loss that one would incur when applying the principle of parallel concatenation to TCM without invoking puncturing. Specifically, this is achieved by puncturing the parity information in a particular manner, so that all information bits are sent only once, and the parity bits are provided alternatively by the two component TCM encoders. The TTCM encoder is shown in Figure 13.11, which comprises two identical TCM encoders linked by a symbol interleaver.

Let the memory of the interleaver be N symbols. The number of modulated symbols per block is $N.n$, where $n = \frac{\mathsf{D}}{2}$ is an integer and D is the number of dimensions of the signal set. The number of information bits transmitted per block is $N.\mathsf{m}$, where m is the number of information bits per symbol. The encoder is clocked at a rate of $n.T$, where T is the symbol duration of each transmitted $2^{(\mathsf{m}+1)/n}$-ary $2\bar{D}$ symbol. At each step, m information bits are input to the TTCM encoder and n symbols each constituted by $\mathsf{m}+1$ bits are transmitted, yielding a coding rate of $\frac{\mathsf{m}}{\mathsf{m}+1}$.

Each component TCM encoder consists of an Ungerböck encoder and a signal mapper. The first TCM encoder operates on the original input bit sequence, while the second TCM encoder manipulates the interleaved version of the input bit sequence. The signal mapper translates the codewords into complex symbols using the SP-based labelling method of Section 13.2.4. A complex symbol represents the amplitude and phase information passed to the modulator in the system seen in Figure 13.11. The complex output symbols of the signal mapper at the bottom of Figure 13.11 are symbol de-interleaved according to the inverse op-

Code	$\tilde{m}$	$H^0(D)$	$H^1(D)$	$H^2(D)$	$H^3(D)$	$d^2_{free}/\triangle^2_0$
$2\bar{D}$, 8PSK, 4 states	2	07	02	04	-	
$2\bar{D}$, 8PSK, 8 states	2	11	02	04	-	3
$4\bar{D}$, 8PSK, 8 states	2	11	06	04	-	3
$2\bar{D}$, 8PSK, 16 states	2	23	02	10	-	3
$4\bar{D}$, 8PSK, 16 states	2	23	14	06	-	3
$2\bar{D}$, 16QAM, 8 states	3	11	02	04	10	2
$2\bar{D}$, 16QAM, 16 states	3	21	02	04	10	3
$2\bar{D}$, 64QAM, 8 states	2	11	04	02	-	3
$2\bar{D}$, 64QAM, 16 states	2	21	04	10	-	4

Table 13.2: 'Punctured' TCM codes exhibiting the best minimum distance for 8PSK, 16QAM and 64QAM, where octal format is used for specifying the generator polynomials [92] ©IEEE, 1998, Robertson and Wörz. The notation $\bar{D}$ denotes the dimensionality of the code while $\triangle^2_0$ denotes the squared Euclidean distance of the signal set itself and d^2_{free} denotes the squared FED of the TCM code.

eration of the interleaver. Again, the interleaver and de-interleaver are symbol interleavers [311]. Owing to invoking the de-interleaver of Figure 13.11 at the output of the component encoder seen at the bottom, the TTCM codewords of both component encoders have identical information bits before the selector. Hence, the selector that alternatively selects the symbols of the upper and lower component encoders is effectively a puncturer that punctures the parity bits of the output symbols.

The output of the selector is then forwarded to the channel interleaver, which is, again, another symbol interleaver. The task of the channel interleaver is to effectively disperse the bursty symbol errors experienced during transmission over fading channels. This increases the diversity order of the code [291, 304]. Finally, the output symbols are modulated and transmitted through the channel.

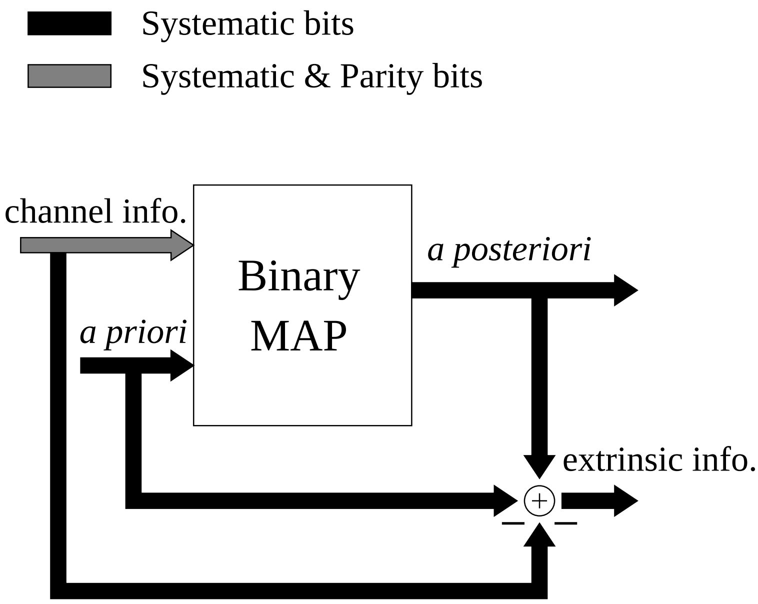
Table 13.2 shows the generator polynomials of some component TCM codes that can be employed in the TTCM scheme. These generator polynomials were obtained by Robertson and Wörz [92] using an exhaustive computer search of all polynomials and finding the one that maximises the minimal Euclidean distance, taking also into account the alternative selection of parity bits for the TTCM scheme. In Table 13.2, $\tilde{m}$ denotes the number of information bits to be encoded out of the total m information bits in a symbol, $\triangle^2_0$ denotes the squared Euclidean distance of the signal set itself, i.e. after TCM signal expansion, and d^2_{free} denotes the squared FED of the TCM constituent codes, as defined in Section 13.2.1. Since $d^2_{free}/\triangle^2_0 > 0$, the 'punctured' TCM codes constructed in Table 13.2 exhibit a positive coding gain in comparison to the uncoded but expanded signal set, although not necessarily in comparison to the uncoded and unexpanded original signal set. Nonetheless, the design target is to provide a coding gain also in comparison to the uncoded and unexpanded original signal set at least for the targeted operational SNR range of the system.

Considering the 8PSK example, where $\triangle^2_0 = d^2_{8PSK}$, we have $d^2_{free}/d^2_{8PSK} = 3$, but when we compare the 'punctured' 8PSK TCM codes with the original uncoded QPSK signal set we have $d^2_{free}/d^2_{QPSK} = d^2_{free}/2 = 0.878$ [92], which implies a negative coding gain. However, when the iterative decoding scheme of TTCM is invoked, we can attain a significant positive coding gain, as we will demonstrate in Section 13.7.

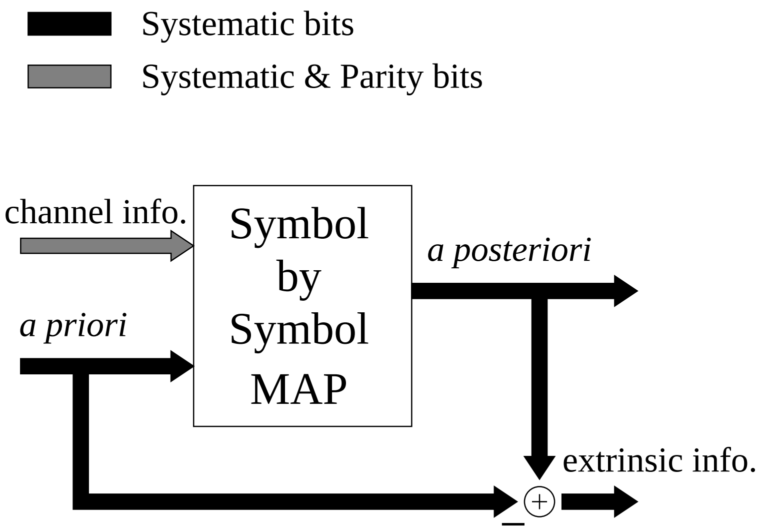
13.4.2 TTCM Decoder

Recall that in Figure 3.9 of Section 3.3.4 the concept of $a-priori$, $a-posteriori$ and $extrinsic$ information was introduced. This illustration is repeated here in Figure 13.12(a) for the sake of convenient comparison with its non-binary counterpart seen in Figure 13.12(b). The associated concept is portrayed in more detail in Figure 13.13, which will be detailed during our further discourse.

The TTCM decoder structure of Figure 13.13(b) is similar to that of binary turbo codes shown in Figure 13.13(a), except that there is a difference in the nature of the information passed from one decoder to the other and in the treatment of the very first decoding step. Specifically, each decoder alternately processes

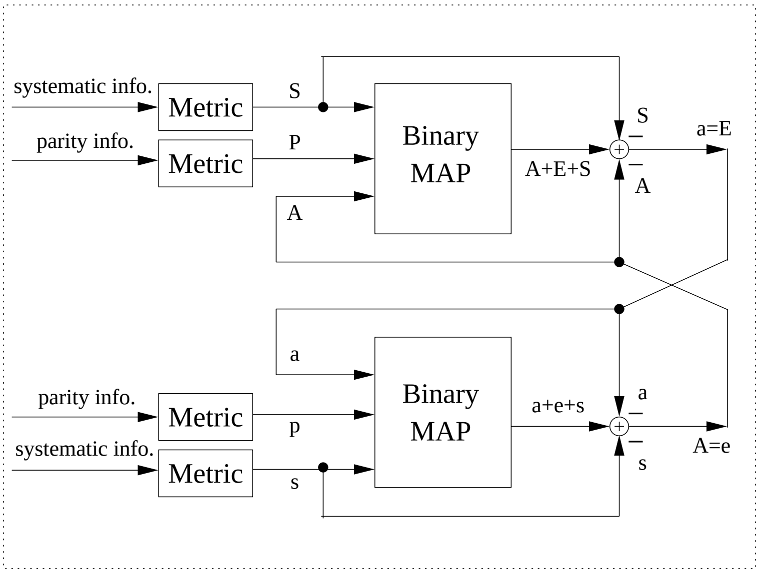


(a) A binary turbo component decoder

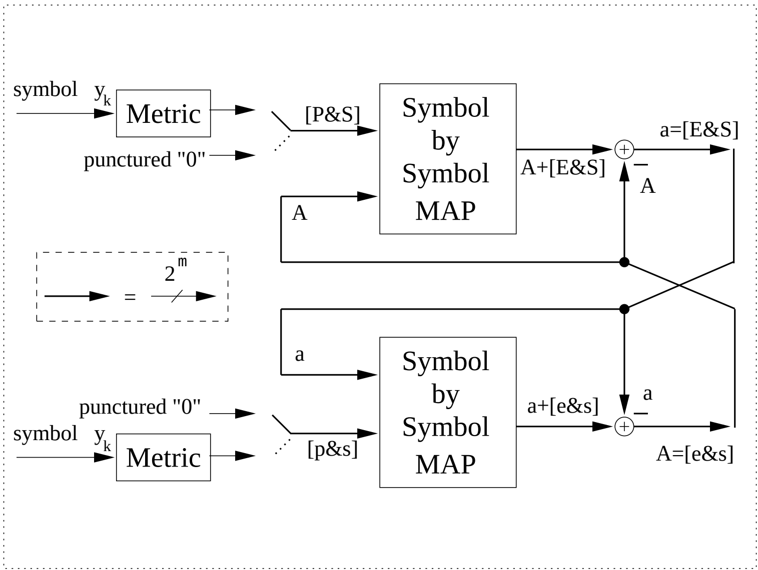


(b) A non-binary TTCM component decoder

Figure 13.12: Schematic of the component decoders for binary turbo codes and non-binary TTCM.



(a) Binary Turbo Decoder at step k



(b) TTCM Decoder at step k

Figure 13.13: Schematic of the decoders for binary turbo codes and TTCM. Note that the labels and arrows apply only to one specific information bit for the binary turbo decoder, or a group of m information bits for the TTCM decoder [92] ©IEEE, 1998, Robertson and Wörz. The interleavers/de-interleavers are not shown and the notations P, S, A and E denote the parity information, systematic information, $a-priori$ probabilities and $extrinsic$ probabilities, respectively. Upper (lower) case letters represent the probabilities of the upper (lower) component decoder.

its corresponding encoder's channel-impaired output symbol, and then the other encoder's channel-impaired output symbol.

In a binary turbo coding scheme the component encoders' output can be split into three additive parts for each information bit u_k at step k, when operating in the logarithmic or LLR domain [52] as shown in Figure 13.13(a), which are:

1) the systematic component (S/s), i.e. the corresponding received systematic value for bit u_k;
2) the $a - priori$ or intrinsic component (A/a), i.e. the information provided by the other component decoder for bit u_k; and
3) the extrinsic information component related to bit u_k (E/e), which depends not on bit u_k itself but on the surrounding bits.

These components are impaired by independent noise and fading effects. In turbo codes, only the extrinsic component should be passed on to the other component decoder, so that the intrinsic information directly related to a bit is not reused in the other component decoder [12]. This measure is necessary in turbo codes for avoiding the prevention of achieving iterative gains, due to the dependence of the constituent decoders' information on each other.

However, in a symbol-based non-binary TTCM scheme the m systematic information and the parity bit are transmitted together in the same non-binary symbol. Hence, the systematic component of the non-binary symbol, namely the original information bits, cannot be separated from the extrinsic component, since the noise and/or fading that affects the parity component also affects the systematic component. Therefore, in this scenario the symbol-based information can be split into only two components:

1) the a-priori component of a non-binary symbol (A/a), which is provided by the other component decoder, and
2) the inseparable extrinsic as well as systematic component of a non-binary symbol $([E\&S]/[e\&s])$, as can be seen from Figure 13.13(b).

Each decoder passes only the latter information to the next component decoder while the a-priori information is removed at each component decoder's output, as seen in Figure 13.13(b), where, again, the extrinsic and systematic components are inseparable.

As described in Section 13.4.1, the number of modulated symbols per block is $N \cdot n$, with $n = \frac{\mathsf{D}}{2}$, where D is the number of dimensions of the signal set. Hence for a $2\bar{D}$ signal set we have $n = 1$ and the number of modulated symbols per block is N. Therefore the symbol interleaver of length N will interleave a block of N complex symbols. Let us consider $2\bar{D}$ modulation having a coding rate of $\frac{\mathsf{m}}{\mathsf{m+1}}$ for the following example.

The received symbols are input to the 'Metric' block of Figure 13.14, in order to generate a set of $\bar{M} = 2^{\mathsf{m}+1}$ symbol probabilities for quantifying the likelihood that a certain symbol of the $\bar{M}$-ary constellation was transmitted. The selector switches seen at the input of the 'Symbol by Symbol MAP' decoder select the current symbol's reliability metric, which is produced at the output of the 'Metric' block, if the current symbol was not punctured by the corresponding encoder. Otherwise puncturing will be applied where the probabilities of the various legitimate symbols at index k are set to 1 or to 0 in the log-domain. The upper (lower) case letters denote the set of probabilities of the upper (lower) component decoder, as shown in the figure. The 'Metric' block provides the decoder with the inseparable parity and systematic $([P\&S]\ or\ [p\&s])$ information, and the second input to the decoder is the a-priori $(A\ or\ a)$ information provided by the other component decoder. The MAP decoder then provides the a-posteriori $(A + [E\&S]\ or\ a + [e\&s])$ information at its output. Then $A\ (or\ a)$ is subtracted from the a-posteriori information, so that the same information is not used more than once in the other component decoder, since otherwise the component decoders' corresponding information would become dependent on each other, which would preclude the achievement of iteration gains. The resulting $[E\&S\ or\ e\&s]$ information is symbol interleaved (or de-interleaved) in order to present the $a\ (or\ A)$ input for the other component decoder in the required order. This decoding process will continue iteratively, in order to offer an improved version of the set of symbol reliabilities for the other component decoder. One iteration comprises the decoding of the received symbols by both the component decoders once. Finally, the a-posteriori information of the lower

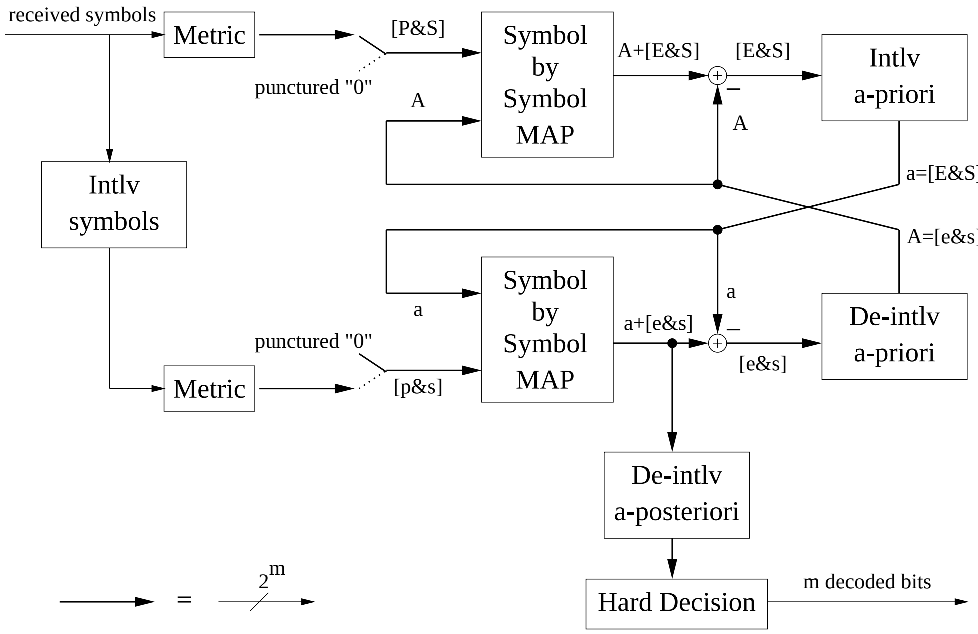


Figure 13.14: Schematic of the TTCM decoder. P, S, A and E denote the parity information, systematic information, $a-priori$ probabilities and $extrinsic$ probabilities, respectively. Upper (lower) case letters represent the probabilities of the upper (lower) component decoder.

component decoder will be de-interleaved in order to extract m decoded information bits per symbol. Hard decision implies selecting the specific symbol which exhibits the maximum a-posteriori probability associated with the m-bit information symbol out of the 2^m probability values. Having described the operation of the symbol-based TTCM technique, which does not protect all transmitted bits of the symbols, let us now consider bit-interleaved coded modulation as a design alternative.

13.5 Bit-interleaved Coded Modulation

Bit-interleaved Coded Modulation (BICM) was proposed by Zehavi [84] with the aim of increasing the diversity order of Ungerböck's TCM schemes which was quantified in Section 13.2.3. Again, the diversity order of a code is defined as the 'length' of the shortest error event path expressed in terms of the number of trellis stages encountered, before remerging with the all-zero path [304] or, equivalently, defined as the minimum Hamming distance of the code [91] where the diversity order of TCM using a symbol-based interleaver is the minimum number of different symbols between the erroneous path and the correct path along the shortest error event path. Hence, in a TCM scenario having parallel transitions, as shown in Figure 13.4, the code's diversity order is one, since the shortest error event path consists of one branch. This implies that parallel transitions should be avoided in TCM codes at all was possible, and if there were no parallel branches, any increase in diversity would be obtained by increasing the constraint length of the code. Unfortunately no TCM codes exist where the parallel transitions associated with the unprotected bits are avoided. In order to circumvent this problem, Zehavi's idea [84] was to render the code's diversity equal to the smallest number of different bits, rather than to that of the different channel symbols, by employing bit-based interleaving, as will be highlighted below.

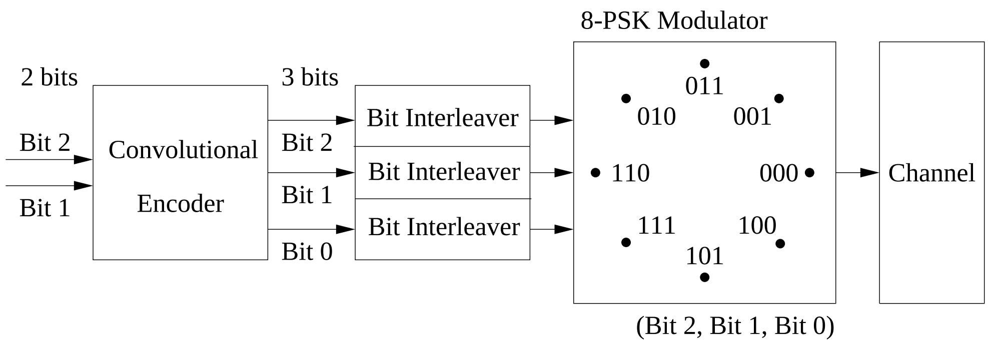


Figure 13.15: BICM encoder schematic employing independent bit interleavers and and protecting all transmitted bits. Instead of the SP-based labelling of TCM in Figure 13.1 here Gray labelling is employed [84] ©IEEE, 1992, Zehavi.

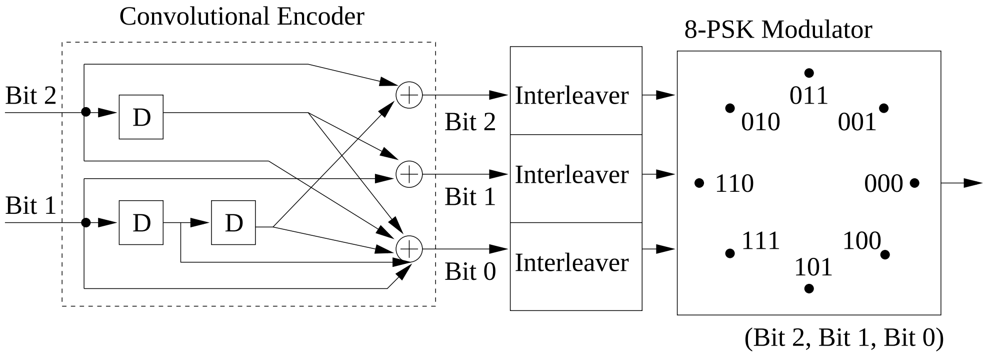


Figure 13.16: Paaske's non-systematic convolutional encoder, bit-based interleavers and modulator forming the BICM encoder [84, 120], where none of the bits are unprotected and instead of the SP-based labelling as seen in Figure 13.1 here Gray labelling is employed.

13.5.1 BICM Principle

The BICM encoder is shown in Figure 13.15. In comparison to the TCM encoder of Figure 13.5, the differences are that BICM uses independent bit interleavers for all the bits of a symbol and non-systematic convolutional codes, rather than a single symbol-based interleaver and systematic RSC codes protecting some of the bits. The number of bit interleavers equals the number of bits assigned to the non-binary codeword. The purpose of bit interleaving is:

- to disperse the bursty errors induced by the correlated fading and to maximise the diversity order of the system;
- to render the bits associated with a given transmitted symbol uncorrelated or independent of each other.

The interleaved bits are then grouped into non-binary symbols, where Gray-coded labelling is used for the sake of optimising the performance of the BICM scheme. The BICM encoder uses Paaske's non-systematic convolutional code proposed on p. 331 of [120], which exhibits the highest possible free Hamming distance, hence attaining optimum performance over Rayleigh fading channels. Figure 13.16 shows Paaske's non-systematic eight-state code of rate-$2/3$, exhibiting a free bit-based Hamming distance of four. The BICM decoder implements the inverse process, as shown in Figure 13.17. In the demodulator module six bit metrics associated with the three bit positions, each having binary values of 0 and 1, are generated from each channel symbol. These bit metrics are de-interleaved by three independent bit de-interleavers, in order

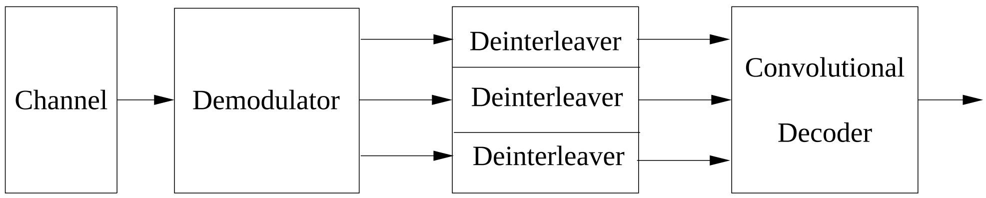


Figure 13.17: BICM decoder [84].

Rate	K	$g^{(1)}$	$g^{(2)}$	$g^{(3)}$	$g^{(4)}$	d_{free}
1/2	3	15	17	-	-	5
	6	133	171	-	-	10
2/3	3	4	2	6	-	4
		1	4	7	-	
	4	7	1	4	-	5
		2	5	7	-	
	6	64	30	64	-	7
		30	64	74	-	
3/4	3	4	4	4	4	4
		0	6	2	4	
		0	2	5	5	
	5	6	2	2	6	5
		1	6	0	7	
		0	2	5	5	
	6	6	1	0	7	6
		3	4	1	6	
		2	3	7	4	

Table 13.3: Paaske's non-systematic convolutional codes, p. 331 of [120], where K denotes the code memory and d_{free} denotes the free Hamming distance. Octal format is used for representing the generator polynomial coefficients.

to form the estimated codewords. Then the convolutional decoder of Figure 13.17 is invoked for decoding these codewords, generating the best possible estimate of the original information bit sequence.

From Equation 13.6 we know that the average bit error probability of a coded modulation scheme using MPSK over Rayleigh fading channels at high SNRs is inversely proportional to $(E_s/N_0)^L$, where E_s/N_0 is the channel's symbol energy to noise spectral density ratio and L is the minimum Hamming distance or the code's diversity order. When bit-based interleavers are employed in BICM instead of the symbol-based interleaver employed in TCM, the minimum Hamming distance of BICM is quantified in terms of the number of different bits between the erroneous path in the shortest error event and the correct path. Since in BICM the bit-based minimum Hamming distance is maximised, BICM will give a lower bit error probability in Rayleigh fading channels than that of TCM maximising the FED. Again, the design of BICM is aimed at providing maximum minimum Hamming distance, rather than providing maximum FED, as in TCM schemes. Moreover, we note that attaining a maximum FED is desired for transmission over Gaussian channels, as shown in Section 13.2.1. Hence, the performance of BICM is not as good as that of TCM in AWGN channels. The reduced FED of BICM is due to the 'random' modulation imposed by the 'random' bit interleavers [84], where the $\bar{\mathsf{m}}$-bit BICM, coded symbol is randomised by the $\bar{\mathsf{m}}$ number of bit interleavers. Again, m denotes the number of information bits, while $\bar{\mathsf{m}}$ denotes the total number of bits in a $2^{\bar{\mathsf{m}}}$-ary modulated symbol.

Table 13.3 summarises the parameters of a range of Paaske's non-systematic codes utilised in BICM. For a rate-k/n code there are k generator polynomials, each having n coefficients. For example, $\mathbf{g_i} =$

Rate	K	$g^{(1)}$	$g^{(2)}$	puncturing matrix	d_{free}
5/6	3	15	17	1 0 0 1 0 0 1 1 1 1	3
	6	133	171	1 1 1 1 1 1 0 0 0 0	3

Table 13.4: Rate-Compatible Punctured Convolutional (RCPC) codes [198, 312], where K denotes the code memory and d_{free} denotes the free Hamming distance. Octal format is used for representing the generator polynomial coefficients.

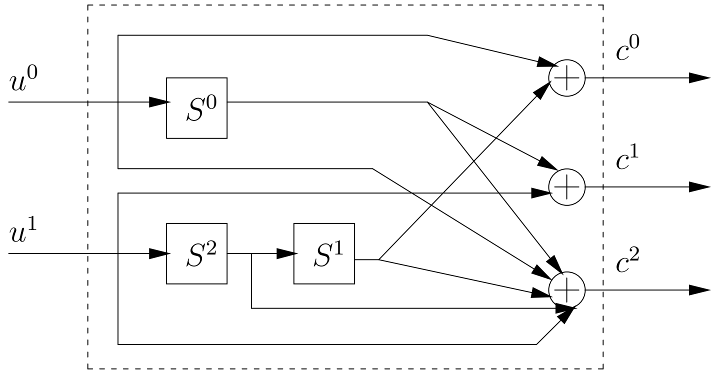


Figure 13.18: Paaske's non-systematic convolutional encoder [120].

$(g^0,\ g^1,\ \ldots,\ g^n),\ i \le k$, is the generator polynomial associated with generating the ith information bit. The generator matrix of the encoder seen in Figure 13.16 is:

$$\mathbf{G(D)} = \begin{bmatrix} 1 & D & 1+D \\ D^2 & 1 & 1+D+D^2 \end{bmatrix}, \tag{13.28}$$

while the equivalent polynomial expressed in octal form is given by:

$$\mathbf{g_1} = \begin{bmatrix} 4 & 2 & 6 \end{bmatrix}\ \mathbf{g_2} = \begin{bmatrix} 1 & 4 & 7 \end{bmatrix}. \tag{13.29}$$

Observe in Table 13.3 that Paaske generated codes of rate-$1/2$, $2/3$ and $3/4$, but not $5/6$. In order to study rate-$5/6$ BICM/64QAM, we created the required punctured code from the rate-$1/2$ code of Table 13.3. Table 13.4 summarises the parameters of the Rate-Compatible Punctured Convolutional (RCPC) codes that can be used in rate=$5/6$ BICM/64QAM schemes. Specifically, rate-$1/2$ codes were punctured according to the puncturing matrix of Table 13.4 in order to obtain the rate-$5/6$ codes, following the approach of [198, 312]. Let us now consider the operation of BICM with the aid of an example.

13.5.2 BICM Coding Example

Considering Paaske's eight-state convolutional code [120] in Figure 13.18 as an example, the BICM encoding process is illustrated here. The corresponding generator polynomial is shown in Equation 13.29. A two-bit information word, namely $u = (u^1,\ u^0)$, is encoded in each cycle in order to form a three-bit codeword, $c = (c^2,\ c^1,\ c^0)$. The encoder has three shift registers, namely S^0, S^1 and S^2, as shown in the figure. The three-bit binary contents of these registers represent eight states, as follows:

$$S = (S^2,\ S^1,\ S^0) \in \{000,\ 001,\ \ldots,\ 111\} = \{0,\ 1,\ \ldots,\ 7\}. \tag{13.30}$$

The input sequence, u, generates a new state S and a new codeword c at each encoding cycle. Table 13.5 illustrates the codewords generated and the associated state transitions. The encoding process can also be

State $S = (S^2, S^1, S^0)$	Information Word $u = (u^1,\ u^0)$			
	$00 = 0$	$01 = 1$	$10 = 2$	$11 = 3$
$000 = 0$	$000 = 0$	$101 = 5$	$110 = 6$	$011 = 3$
$001 = 1$	$110 = 6$	$011 = 3$	$000 = 0$	$101 = 5$
$010 = 2$	$101 = 5$	$000 = 0$	$011 = 3$	$110 = 6$
$011 = 3$	$011 = 3$	$110 = 6$	$101 = 5$	$000 = 0$
$100 = 4$	$100 = 4$	$001 = 1$	$010 = 2$	$111 = 7$
$101 = 5$	$010 = 2$	$111 = 7$	$100 = 4$	$001 = 1$
$110 = 6$	$001 = 1$	$100 = 4$	$111 = 7$	$010 = 2$
$111 = 7$	$111 = 7$	$010 = 2$	$001 = 1$	$100 = 4$
Codeword $c = (c^2,\ c^1, c^0)$				
$000 = 0$	$000 = 0$	$001 = 1$	$100 = 4$	$101 = 5$
$001 = 1$	$000 = 0$	$001 = 1$	$100 = 4$	$101 = 5$
$010 = 2$	$000 = 0$	$001 = 1$	$100 = 4$	$101 = 5$
$011 = 3$	$000 = 0$	$001 = 1$	$100 = 4$	$101 = 5$
$100 = 4$	$010 = 2$	$011 = 3$	$110 = 6$	$111 = 7$
$101 = 5$	$010 = 2$	$011 = 3$	$110 = 6$	$111 = 7$
$110 = 6$	$010 = 2$	$011 = 3$	$110 = 6$	$111 = 7$
$111 = 7$	$010 = 2$	$011 = 3$	$110 = 6$	$111 = 7$
Next State $S = (S^2,\ S^1,\ S^0)$				

Table 13.5: The codeword generation and state transition table of the non-systematic convolutional encoder of Figure 13.18. The state transition diagram is seen in Figure 13.19.

represented with the aid of the trellis diagram of Figure 13.19. Specifically, the top part of Table 13.19 contains the codewords $c = (c^2,\ c^1, c^0)$ as a function of the encoder state $S = (S^2, S^1, S^0)$ as well as that of the information word $u = (u^1,\ u^0)$, while the bottom section contains the next states, again as a function of S and u. For example, if the input is $u = (u^1,\ u^0) = (1,1) = 3$ when the shift register is in state $S = (S^2, S^1, S^0) = (1,1,0) = 6$, the shift register will change its state to state $S = (S^2, S^1, S^0) = (1,1,1) = 7$ and $c = (c^2,\ c^1, c^0) = (0,1,0) = 2$ will be the generated codeword. Hence, if the input binary sequence is $\{01\ 10\ 01\ 00\ 10\ 10 \rightarrow\}$ with the rightmost being the first input bit, the corresponding information words are $\{1\ 2\ 1\ 0\ 2\ 2 \rightarrow\}$. Before any decoding takes place, the shift register is initialised to zero. Therefore, as seen at the right of Figure 13.19, when the first information word of $u_1 = 2$ arrives, the state changes from $S^{-1} = 0$ to $S = 4$, generating the first codeword $c_1 = 6$ as seen in the bottom and top sections of Table 13.5, respectively. Then the second information word of $u_2 = 2$ changes the state from $S^{-1} = 4$ to $S = 6$, generating the second codeword of $c_2 = 2$. The process continues in a similiar manner according to the transition table, namely Table 13.5. The codewords generated as seen at the right of Figure 13.19 are $\{4\ 0\ 0\ 1\ 2\ 6 \rightarrow\}$, and the state transitions are $\{2 \leftarrow 4 \leftarrow 1 \leftarrow 2 \leftarrow 6 \leftarrow 4 \leftarrow 0\}$. Then the bits constituting the codeword sequence are interleaved by the three bit interleavers of Figure 13.16, before they are assigned to the corresponding 8PSK constellation points.

13.6 Bit-Interleaved Coded Modulation Using Iterative Decoding

BICM using Iterative Decoding (BICM-ID) was proposed by Li [90, 293] for further improving the FED of Zehavi's BICM scheme, although BICM already improved the diversity order of Ungerböck's TCM scheme. This FED improvement can be achieved with the aid of combining SP-based constellation labelling, as in TCM, and by invoking soft-decision feedback from the decoder's output to the demodulator's input, in order to exchange soft-decision-based information between them. As we will see below, this is advantageous, since upon each iteration the channel decoder improves the reliability of the soft information passed to the

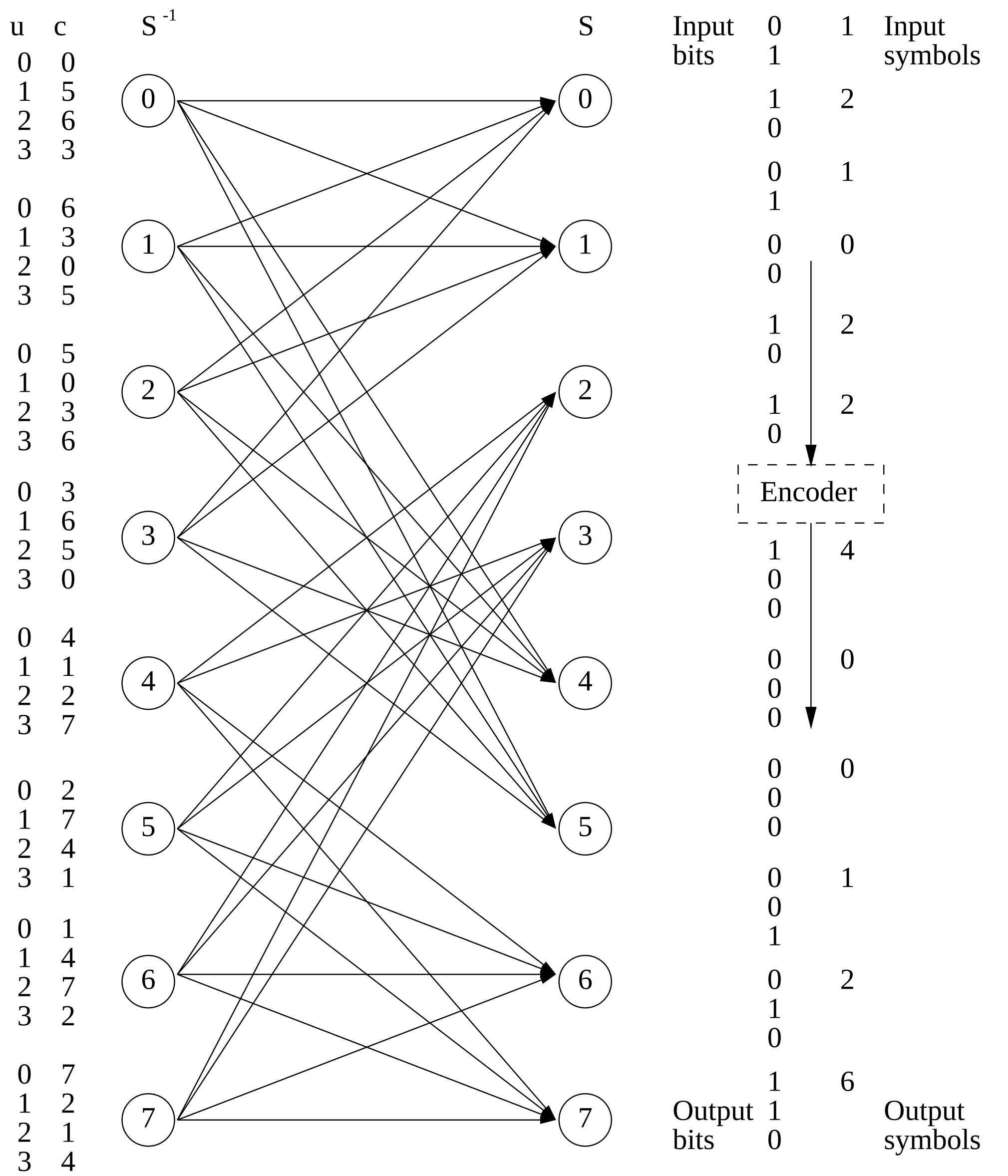


Figure 13.19: Trellis diagram for Paaske's eight-state convolutional code, where u indicates the information word, c indicates the codeword, S^{-1} indicates the previous state and S indicates the current state. As an example, the encoding of the input bit sequence of $\{011001001010 \rightarrow\}$ is shown at the right. The encoder schematic is portrayed in Figure 13.18, while the state transitions are summarised in Table 13.5.

demodulator.

13.6.1 Labelling Method

Let us now consider the mapping of the interleaved bits to the phasor constellation in this section. Figure 13.20 shows the process of subset partitioning for each of the three bit positions for both Gray labelling

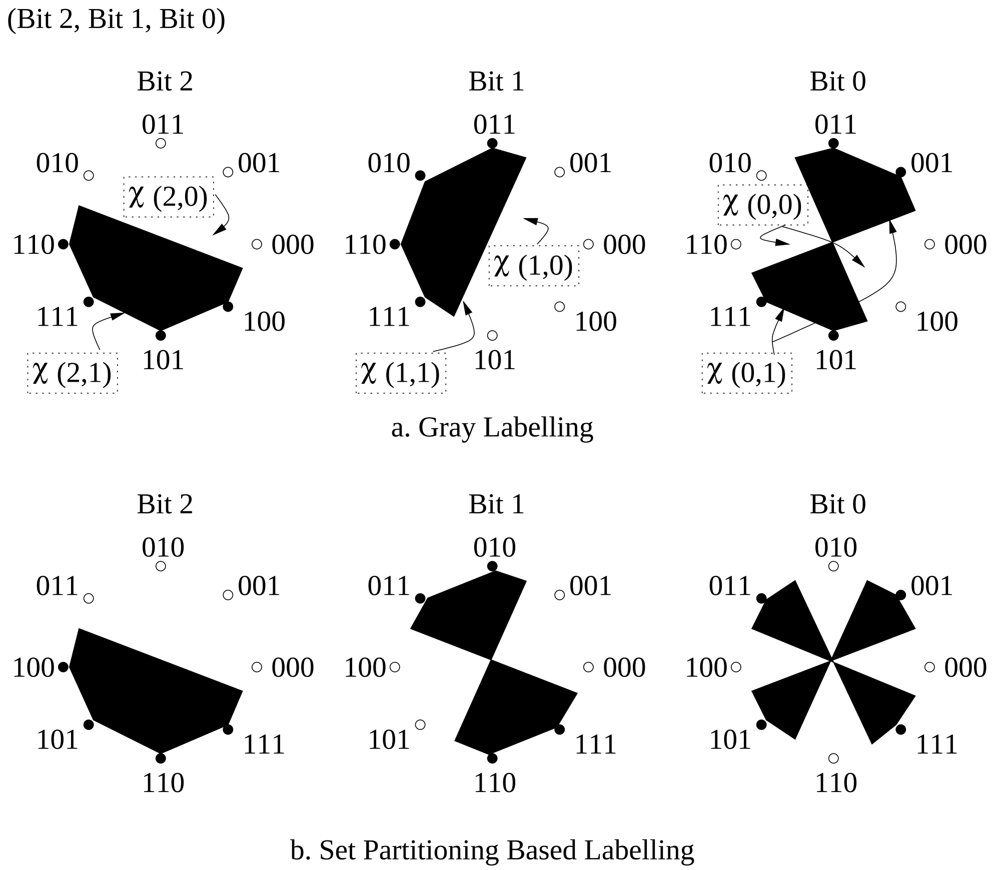


Figure 13.20: SP and Gray labelling methods for 8PSK and the corresponding subset partitioning for each bit, where $\chi(i,b)$ defined in Equation 13.36 refers to the subset of the modulation constellation for Bit i where Bit $i = b \in \{0,1\}$ [293] ©IEEE, 1999, Li and Ritcey.

and in the context of SP labelling. The shaded regions shown inside the circle correspond to the subset $\chi(i,1)$ defined in Equation 13.36, and the unshaded regions to $\chi(i,0)$, $i = 0,1,2$, where i indicates the bit position in the three-bit BICM/8PSK symbol. These are also the decision regions for each bit, if hard-decision-based BICM demodulation is used for detecting each bit individually. The two labelling methods seen in Figure 13.20 have the same intersubset distances, although a different number of nearest neighbours. For example, $\chi(0,1)$, which denotes the region where bit 0 equals to 1, is divided into two regions in the context of Gray labelling, as can be seen in Figure 13.20(a). By contrast, in the context of SP labelling seen in Figure 13.20(b), $\chi(0,1)$ is divided into four regions. Clearly, Gray labelling has a lower number of nearest neighbours compared to SP-based labelling. The higher the number of nearest neighbours, the higher the chances for a bit to be decoded into the wrong region. Hence, Gray labelling is a more appropriate mapping during the first decoding iteration, and hence it was adopted by the non-iterative BICM scheme of Figure 13.17.

During the second decoding iteration in BICM-ID, given the feedback information representing the original uncoded information bits in Figure 13.16, namely Bit 1 and Bit 2, the constellation associated with Bit 0 is confined to a pair of constellation points, as shown at the right of Figure 13.21. Therefore, as far as Bit 0 is concerned, the 8PSK phasor constellation is translated into four binary constellations, where one of the four possible specific BPSK constellations is selected by the feedback Bit 1 and Bit 2. The same is true for the constellations associated with both Bit 1 and Bit 2, given the feedback information of the

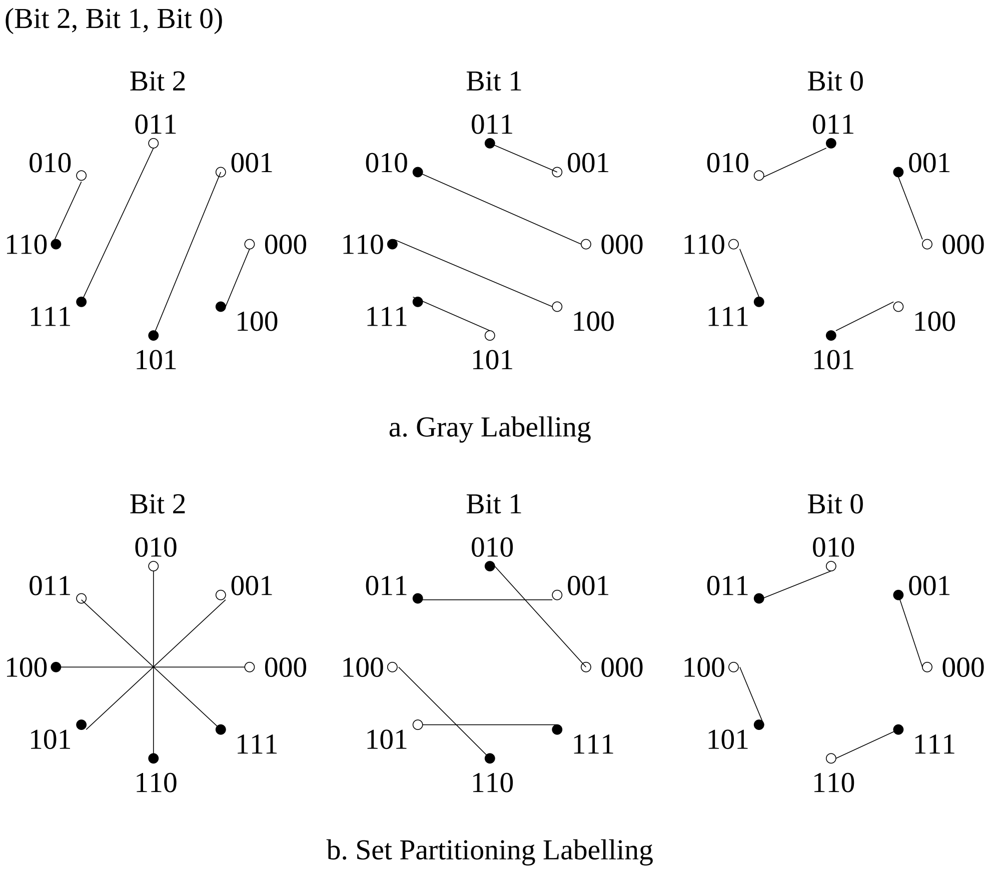


Figure 13.21: Iterative decoding translates the 8PSK scheme into three parallel binary sub-channels, each associated with a BPSK constellation selected from the four possible signal sets [293] ©IEEE, 1999, Li and Ritcey.

corresponding other two bits.

In order to optimise the second-pass decoding performance of BICM-ID, one must maximise the minimum Euclidean distance between any two points of all the $2^{\bar{m}-1} = 4$ possible phasor pairs at the left (Bit 2), centre (Bit 1) and the right (Bit 0) of Figure 13.21. Clearly, SP-based labelling serves this aim better, when compared to Gray labelling, since the corresponding minimum Euclidean distance of SP-based labelling is higher than that of Gray labelling for both Bit 1 and Bit 2, as illustrated at the left and the centre of Figure 13.21. Although the first-pass performance is important, in order to prevent error precipitation due to erroneous feedback bits, the error propagation is effectively controlled by the soft feedback of the decoder. Therefore, BICM-ID assisted by soft decision feedback uses SP labelling.

Specifically, the desired high Euclidean distance for Bit 2 in Figure 13.21(b) is only attainable when Bit 1 and Bit 0 are correctly decoded and fed back to the SP-based demodulator. If the values to be fed back are not correctly decoded, the desired high Euclidean distance will not be achieved and error propagation will occur. On the other hand, an optimum convolutional code having a high binary Hamming distance is capable of providing a high reliability for the decoded bits. Therefore, an optimum convolutional code using appropriate signal labelling is capable of 'indirectly' translating the high binary Hamming distance between coded bits into a high Euclidean distance between the phasor pairs portrayed in Figure 13.21. In short, BICM-ID converts a $2^{\bar{m}}$-ary signalling scheme to $\bar{m}$ independent parallel binary schemes by the employment of $\bar{m}$ number of independent bit interleavers and involves an iterative decoding method. This

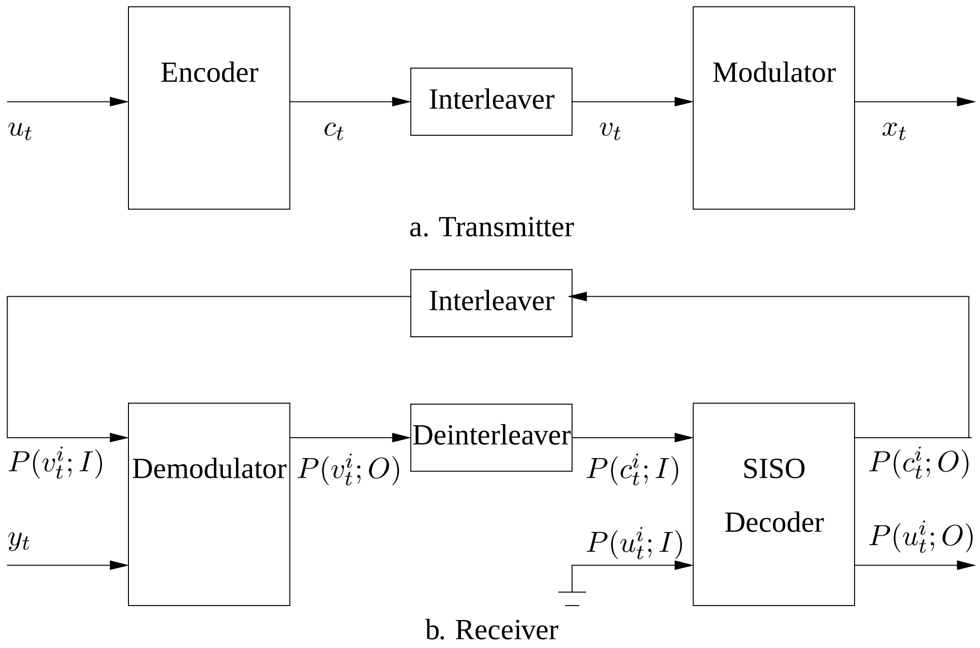


Figure 13.22: The transmitter and receiver modules of the BICM-ID scheme using soft-decision feedback [90] ©IEEE, 1998, Li.

simultaneously facilitates attaining a high diversity order with the advent of the bit interleavers, as well as achieving a high FED with the aid of the iterative decoding and SP-based labelling. Hence, BICM-ID effectively combines powerful binary codes with bandwidth-efficient modulation.

13.6.2 Interleaver Design

The interleaver design is important as regards the performance of BICM-ID. In [294], Li introduced certain constraints on the design of the interleaver, in order to maximise the minimum Euclidean distance between the two points in the $2^{\bar{m}-1}$ possible specific BPSK constellations. However, we advocate a more simple approach, where the $\bar{m}$ number of interleavers used for the $2^{\bar{m}}$-ary modulation scheme are generated randomly and separately, without any interactions between them. The resultant minimum Euclidean distance is less than that of the scheme proposed in [294], but the error bursts inflicted by correlated fading are expected to be randomised effectively by the independent bit interleavers. This was expected to give a better performance over fading channels at the cost of a slight performance degradation over AWGN channels, when compared to Li's scheme [294]. However, as we will demonstrate in the context of our simulation results in Section 13.7.2.2, our independent random interleaver design and Li's design perform similarly.

Having described the labelling method and the interleaver design in the context of BICM-ID, let us now consider the operation of BICM-ID with the aid of an example.

13.6.3 BICM-ID Coding Example

The BICM-ID scheme using soft-decision feedback is shown in Figure 13.22. The interleavers used are all bit-based, as in the BICM scheme of Figure 13.16, although for the sake of simplicity here only one interleaver is shown. A Soft-Input Soft-Output (SISO) [313] decoder is used in the receiver module and the decoder's output is fed back to the input of the demodulator. The SISO decoder of the BICM-ID scheme is actually a MAP decoder that computes the a-posteriori probabilities for the non-systematically channel-coded bits and the original information bits.

For an $(n,\ k)$ binary convolutional code the encoder's input symbol at time t is denoted by $u_t = [u_t^0,\ u_t^1,\ \ldots,\ u_t^{k-1}]$ and the coded output symbol by $c_t = [c_t^0,\ c_t^1,\ \ldots,\ c_t^{n-1}]$, where u_t^i or c_t^i is the ith bit in a symbol as defined in the context of Table 13.5 and Figure 13.19. The coded bits are interleaved by $\bar{\mathsf{m}}$ independent bit interleavers, then $\bar{\mathsf{m}}$ interleaved bits are grouped together in order to form a channel symbol $v_t = [v_t^0,\ v_t^1,\ \ldots,\ v^{\bar{\mathsf{m}}-1}]$ as seen in Figure 13.22(a), for transmission using $2^{\bar{\mathsf{m}}}$-ary modulation. Let us consider 8PSK modulation, i.e. $\bar{\mathsf{m}} = 3$ as an example.

A signal labelling method μ maps the symbol v_t to a complex phasor according to $x_t = \mu(v_t)$, $x_t \in \chi$, where the 8PSK signal set is defined as $\chi = \{\sqrt{E_s}\ e^{j2n\pi/8},\ n = 0, \ldots, 7\}$ and E_s is the energy per transmitted symbol. In conjunction with a rate-$2/3$ code, the energy per information bit is $E_b = E_s/2$. For transmission over Rayleigh fading channels using coherent detection, the received discrete time signal is:

$$y_t = \rho_t x_t + n_t, \tag{13.31}$$

where ρ_t is the Rayleigh-distributed fading amplitude [49] having an expectation value of $\mathrm{E}(\rho_t^2) = 1$, while n_t is the complex AWGN exhibiting a variance of $\sigma^2 = N_0/2$ where N_0 is the noise's PSD. For AWGN channels we have $\rho_t = 1$ and the Probability Density Function (PDF) of the non-faded but noise-contaminated received signal is expressed as [306]:

$$P(y_t|x_t, \rho_t) = \frac{1}{2\pi\sigma^2}\ e^{-\frac{1}{2}\left(\frac{n_t}{\sigma}\right)^2}, \tag{13.32}$$

where $\sigma^2 = N_0/2$ and the constant multiplicative factor of $\frac{1}{2\pi\sigma^2}$ does not influence the shape of the distribution and hence can be ignored when calculating the branch transition metric η, as described in Section 13.3.3. For AWGN channels, the conditional PDF of the received signal can be written as:

$$P(y_t|x_t) = e^{-\frac{|y_t\ -\ x_t|^2}{2\sigma^2}}. \tag{13.33}$$

Considering AWGN channels, the demodulator of Figure 13.22(b) takes y_t as its input for computing the confidence metrics of the bits using the maximum APP criterion [298]:

$$P(v_t^i = b|y_t) \quad = \sum_{x_t \in \chi(i,b)} P(x_t|y_t), \tag{13.34}$$

where $i \in \{0, 1, 2\}$, $b \in \{0, 1\}$ and $x_t = \mu(v_t)$. Furthermore, the signal after the demodulator of Figure 13.22 is described by the demapping of the bits $[\nabla^0(x_t), \nabla^1(x_t), \nabla^2(x_t)]$ where $\nabla^i(x_t) \in \{0, 1\}$ is the value of the ith bit of the three-bit label assigned to x_t. With the aid of Bayes' rule in Equation 3.12 we obtain:

$$P(v_t^i = b|y_t) \quad = \sum_{x_t \in \chi(i,b)} P(y_t|x_t)P(x_t), \tag{13.35}$$

where the subset $\chi(i, b)$ is described as:

$$\chi(i, b) = \{\mu([\nabla^0(x_t), \nabla^1(x_t), \nabla^2(x_t)])\ \mid\ \nabla^j(x_t) \in \{0, 1\}, j \neq i\}, \tag{13.36}$$

which contains all the phasors for which $\nabla^i(x_t) = b$ holds. For 8PSK, where $\bar{\mathsf{m}} = 3$, the size of each such subset is $2^{\bar{\mathsf{m}}-1} = 4$ as portrayed in Figure 13.20. This implies that only the a-priori probabilities of $\bar{\mathsf{m}} - 1 = 2$ bits out of the total of $\bar{\mathsf{m}} = 3$ bits per channel symbol have to be considered, in order to compute the bit metric of a particular bit.

Now using the notation of Benedetto *et al.* [313], the a-priori probabilities of an original uncoded information bit at time index t and bit index i, namely u_t^i being 0 and 1, are denoted by $P(u_t^i = 0; I)$ and $P(u_t^i = 1; I)$ respectively, while I refers to the a-prIori probabilities of the bit. This notation is simplified to $P(u_t^i; I)$, when no confusion arises, as shown in Figure 13.22. Similarly, $P(c_t^i; I)$ denotes the a-priori probabilities of a legitimate coded bit at time index t and position index i. Finally, $P(u_t^i; O)$ and $P(c_t^i; O)$ denote the extrinsic a-pOsteriori information of the original information bits and coded bits, respectively.

The a-priori probability $P(x_t)$ in Equation 13.35 is unavailable during the first-pass decoding, hence an equal likelihood is assumed for all the $2^{\bar{m}}$ legitimate symbols. This renders the extrinsic a-posteriori bit probabilities, $P(v_t^i = b; O)$, equal to $P(v_t^i = b|y_t)$, when ignoring the common constant factors. Then, the SISO decoder of Figure 13.22(b) is used for generating the extrinsic a-posteriori bit probabilities $P(u_t^i; O)$ of the information bits, as well as the extrinsic a-posteriori bit probabilities $P(c_t^i; O)$ of the coded bits, from the de-interleaved probabilities $P(v_t^i = b; O)$, as seen in Figure 13.22(b). Since $P(u_t^i; I)$ is unavailable, it is not used in the entire decoding process.

During the second iteration $P(c_t^i; O)$ is interleaved and fed back to the input of the demodulator in the correct order in the form of $P(v_t^i; I)$, as seen in Figure 13.22(b). Assuming that the probabilities $P(v_t^0; I)$, $P(v_t^1; I)$ and $P(v_t^2; I)$ are independent by the employment of three independent bit interleavers, we have for each $x_t \in \chi$:

$$\begin{aligned} P(x_t) &= P(\mu([\nabla^0(x_t), \nabla^1(x_t), \nabla^2(x_t)])) \\ &= \prod_{j=0}^{2} P(v_t^j = \nabla^j(x_t); I), \end{aligned} \tag{13.37}$$

where $\nabla^j(x_t) \in \{0, 1\}$ is the value of the jth bit of the three-bit label for x_t. Now that we have the a-priori probability $P(x_t)$ of the transmitted symbol x_t, the extrinsic a-posteriori bit probabilities for the second decoding iteration can be computed using Equations 13.35 and 13.37, yielding:

$$\begin{aligned} P(v_t^i = b; O) &= \frac{P(v_t^i = b|y_t)}{P(v_t^i = b; I)} \\ &= \sum_{x_t \in \chi(i,b)} \left(P(y_t|x_t) \prod_{j \neq i} P(v_t^j = \nabla^j(x_t); I) \right) \\ & \quad i \in \{0, 1, 2\}, \quad b \in \{0, 1\}. \end{aligned} \tag{13.38}$$

As seen from Equation 13.38, in order to recalculate the metric for a bit we only need the a-priori probabilities of the other two bits in the same channel symbol. After interleaving in the feedback loop of Figure 13.22, the regenerated bit metrics are tentatively soft demodulated again and the process of passing information between the demodulator and decoder is continued. The final decoded output is the hard-decision-based $extrinsic$ bit probability $P(u_t^i; O)$.

So far in Sections 13.2–13.6 we have studied the conceptual differences between four coded modulation schemes in terms of their coding structure, signal labelling philosophy, interleaver type and decoding philosophy. The symbol-based non-binary MAP algorithm was also highlighted, when operating in the log-domain. In the next section we will proceed to study the performance of TCM, BICM, TTCM and BICM-ID when communicating over both narrowband and wideband channels.

13.7 Coded Modulation Performance

13.7.1 Introduction

Having described the principles of TCM, BICM, TTCM and BICM-ID in Sections 13.2–13.6, in this section their performance will be evaluated for transmission over both narrowband and wideband fading channels. Specifically, in Section 13.7.2 we will evaluate the performance of these coded modulation schemes for transmissions over narrowband channels, while in Section 13.7.3 we will consider their performance in the context of wideband channels.

Owing to the Inter-Symbol Interference (ISI) inflicted by wideband channels, the employment of equalisers is essential in assisting the operation of the coded modulation schemes considered. Hence a Decision Feedback Equaliser (DFE) is introduced in Section 13.7.3.2, while Section 13.7.3.3 will evaluate the performance of a DFE-aided wideband burst-by-burst adaptive coded modulation system. Another

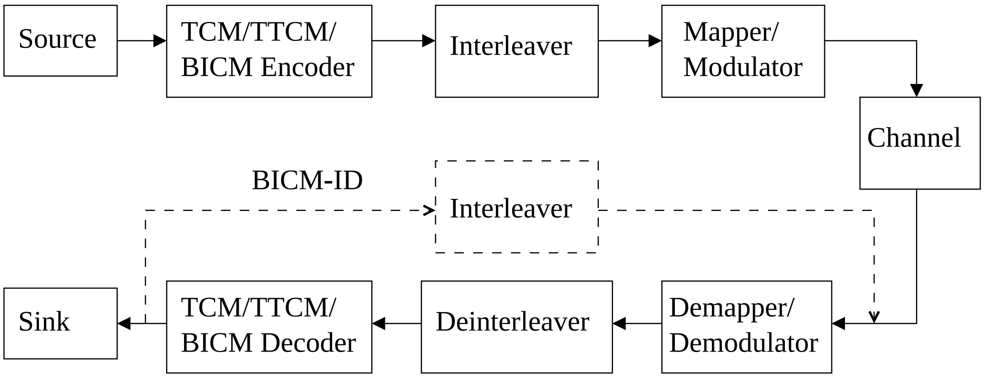


Figure 13.23: System overview of different coded modulation schemes.

approach to overcoming the ISI in wideband channels is the employment of a multi-carrier Orthogonal Frequency Division Multiplexing (OFDM) system. Hence, OFDM is studied in Section 13.7.3.4, while Section 13.7.3.5 evaluates the performance of an OFDM-assisted coded modulation scheme.

13.7.2 Coded Modulation in Narrowband Channels

In this section, a comparative study of TCM, TTCM, BICM and BICM-ID schemes over both Gaussian and uncorrelated narrowband Rayleigh fading channels is presented in the context of eight-level Phase Shift Keying (PSK), 16-level Quadrature Amplitude Modulation (QAM) and 64QAM. We comparatively study the associated decoding complexity, the effects of the encoding block length and the achievable bandwidth efficiency. It will be shown that TTCM constitutes the best compromise scheme, followed by BICM-ID.

13.7.2.1 System Overview

The schematic of the coded modulation schemes under consideration is shown in Figure 13.23. The source generates random information bits, which are encoded by one of the TCM, TTCM or BICM encoders. The coded sequence is then appropriately interleaved and used for modulating the waveforms according to the symbol mapping rules. For a narrowband Rayleigh fading channel in conjunction with coherent detection, the relationship between the transmitted discrete time signal x_t and the received discrete time signal y_t is given by:

$$y_t = \rho_t x_t \ + \ n_t, \tag{13.39}$$

where ρ_t is the Rayleigh-distributed fading amplitude having an expected value of $\mathrm{E}(\rho_t^2) = 1$, while n_t is the complex AWGN having a variance of $\sigma^2 = N_0/2$ where N_0 is the noise's PSD. For AWGN channels we have $\rho_t = 1$. The receiver consists of a coherent demodulator followed by a de-interleaver and one of the TCM, TTCM or BICM decoders. TTCM schemes consist of two component TCM encoders and two parallel decoders. In BICM-ID schemes the decoder output is appropriately interleaved and fed back to the demodulator input, as shown in Figure 13.23.

The log-domain branch metric required for the maximum likelihood decoding of TCM and TTCM over fading channels is given by the squared Euclidean distance between the faded transmitted symbol x_t and the noisy received symbol y_t, which is formulated as:

$$\pi_t = |y_t - \rho_t x_t|^2. \tag{13.40}$$

By contrast, the corresponding branch metric for BICM and BICM-ID is formed by summing the de-interleaved bit metrics λ of each coded bit v_t^i which quantifies the reliability of the corresponding symbol,

Rate	State	$\tilde{\mathsf{m}}$	H^0	H^1	H^2	H^3
2/3	8	2	11	02	04	-
(8PSK)	64 *	2	103	30	66	-
3/4	8	3	11	02	04	10
(16QAM)	64 *	3	101	16	64	-
5/6	8	2	11	02	04	-
(64QAM)	64 *	2	101	16	64	-

Table 13.6: 'Punctured' TCM codes with best minimum distance for PSK and QAM, ©Robertson and Wörz [92]. '*' indicates Ungerböck's TCM codes [290]. Two-dimensional ($2\bar{D}$) modulation is utilised. Octal format is used for representing the generator polynomials H^i and $\tilde{\mathsf{m}}$ denotes the number of coded information bits out of the total m information bits in a modulated symbol.

yielding:

$$\pi_t = \sum_{i=0}^{\mathsf{m}} \lambda(v_t^i = b), \tag{13.41}$$

where i is the bit position of the coded bit in a constellation symbol, m is the number of information bits per symbol and $b \in (0, 1)$. The number of coded bits per symbol is $(\mathsf{m} + 1)$, since the coded modulation schemes add one parity bit to the m information bits by doubling the original constellation size, in order to maintain the same spectral efficiency of m bits/s/Hz. The BICM bit metrics $\tilde{\lambda}$ before the de-interleaver are defined as [293]:

$$\tilde{\lambda}(v_t^i = b) = \sum_{x \in \chi(i,b)} |y_t - \rho_t x|^2, \tag{13.42}$$

where $\chi(i, b)$ is the signal set, for which the bit i of the symbol has a binary value b.

To elaborate a little further, the coded modulation schemes that we comparatively studied are Ungerböck's TCM [290], Robertson's TTCM [92], Zehavi's BICM [84] and Li's BICM-ID [298]. Table 13.6 shows the generator polynomials of both the TCM and TTCM codes in octal format. These are RSC codes that add one parity bit to the information bits. Hence, the coding rate of a $2^{\mathsf{m}+1}$-ary PSK or QAM signal is $R = \frac{\mathsf{m}}{\mathsf{m}+1}$. The number of decoding states associated with a code of memory K is 2^K. When the number of protected/coded information bits $\tilde{\mathsf{m}}$ is less than the total number of original information bits m, there are $(\mathsf{m} - \tilde{\mathsf{m}})$ uncoded information bits and $2^{\mathsf{m}-\tilde{\mathsf{m}}}$ parallel transitions in the trellis of the code. Parallel transitions assist in reducing the decoding complexity and the memory required, since the dimensionality of the corresponding trellis is smaller than that of a trellis having no parallel branches.

Table 13.7 shows the generator polynomials for the BICM and BICM-ID codes in octal format. These codes are non-systematic convolutional codes having a maximum free Hamming distance. Again, only one extra bit is added to the information bits. Hence, the achievable coding rate and the bandwidth efficiency are similar to that of TCM and TTCM for the $2^{\mathsf{m}+1}$-ary modulation schemes used. In order to reduce the required decoding memory, the BICM and BICM-ID schemes based on 64QAM were obtained by puncturing the rate-1/2 codes following the approach of [312], since for a non-punctured rate-5/6 code there are $2^{(\mathsf{m}=5)} = 32$ branches emerging from each trellis state for a block length of $\bar{L}$, whereas for the punctured rate-1/2 code, there are only $2^{(\mathsf{m}=1)} = 2$ branches emerging from each trellis state for a block length of $\mathsf{m}\bar{L} = 5\bar{L}$. Therefore the required decoding memory is reduced by a factor of $\frac{2^{\mathsf{m}} \cdot \bar{L}}{2^1 \cdot \mathsf{m} \cdot \bar{L}} = 3.2$.

Soft-decision trellis decoding utilising the Log-Maximum A-Posteriori (Log-MAP) algorithm [52] was invoked for the decoding of the coded modulation schemes. As discussed in Section 13.3.3, the Log-MAP algorithm is a numerically stable version of the MAP algorithm operating in the log-domain, in order to reduce its complexity and to mitigate the numerical problems associated with the MAP algorithm [11].

Rate	State	g^1	g^2	g^3	g^4	d_{free}
2/3	8	4	2	6	-	4
(8PSK)	(K=3)	1	4	7	-	
	16	7	1	4	-	5
	(K=4)	2	5	7	-	
	64	15	6	15	-	7
	(K=6)	6	15	17	-	
3/4	8	4	4	4	4	4
(16QAM)	(K=3)	0	6	2	4	
		0	2	5	5	
	32	6	2	2	6	5
	(K=5)	1	6	0	7	
		0	2	5	5	

Rate	State	g^1	g^2	Puncturing	d_{free}
5/6	8	15	17	1 0 0 1 0	3
(64QAM)	(K=3)			0 1 1 1 1	
	64	133	171	1 1 1 1 1	3
	(K=6)			1 0 0 0 0	

Table 13.7: Top table shows the generator polynomials of Paaske's code, p. 331 of [120]. Bottom table shows those of the rate-compatible puncture convolutional codes [312]. K is the code memory and d_{free} is the free Hamming distance. Octal format is used for the polynomial coefficients g^i, while '1' and '0' in the puncturing matrix indicate the position of the unpunctured and punctured coded bits, respectively.

13.7.2.2 Simulation Results and Discussions

In this section we study the performance of TCM, TTCM, BICM and BICM-ID using computer simulations. The complexity of the coded modulation schemes is compared in terms of the number of decoding states, and the number of decoding iterations. For a TCM or BICM code of memory K, the corresponding complexity is proportional to the number of decoding states namely to $S = 2^K$. Since TTCM schemes invoke two component TCM codes, a TTCM code with t iterations and using an S-state component code exhibits a complexity proportional to $2.t.S$ or $t.2^{K+1}$. As for BICM-ID schemes, only one decoder is used but the demodulator is invoked in each decoding iteration. However, the complexity of the demodulator is assumed to be insignificant compared to that of the channel decoder. Hence, a BICM-ID code with t iterations using an S-state code exhibits a complexity proportional to $t.S$ or $t.2^K$.

13.7.2.2.1 Coded Modulation Performance over AWGN Channels It is important to note that in terms of the total number of trellis states the decoding complexity of 64-state TCM and 8-state TTCM using two TCM decoders in conjunction with four iterations can be considered similar. The same comments are valid also for 16-state BICM-ID using four iterations or for 8-state BICM-ID using eight iterations. In our forthcoming discourse we will always endeavour to compare schemes of similar decoding complexity, unless otherwise stated. Figure 13.24 illustrates the effects of interleaving block length on the TCM, TTCM and BICM-ID performance in an 8PSK scheme over AWGN channels. It is clear from the figure that a high interleaving block length is desired for the iterative TTCM and BICM-ID schemes. The block length does not affect the BICM-ID performance during the first pass, since it constitutes a BICM scheme using SP-based phasor labelling. However, if we consider four iterations, the performance improves, converging faster to the Error-Free-Feedback (EFF) bound[2] [293] for larger block lengths. At a BER of 10^{-4} a 500-bit block length was about 1 dB inferior in terms of the required SNR to the 2000-bit block length in the

[2] The EFF bound is defined as the BER upper bound performance achieved for the idealised situation, when the decoded values fed back to the demodulator in Figure 13.23 are error free.

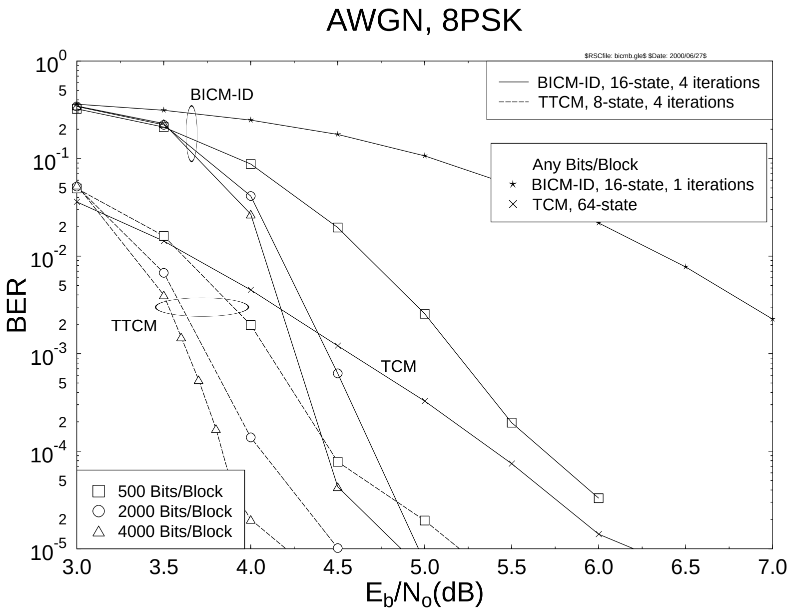


Figure 13.24: Effects of block length on the TCM, TTCM and BICM-ID performance in the context of an 8PSK scheme for transmissions over AWGN channels.

context of the BICM-ID scheme. A slight further SNR improvement was obtained for the 4000-bit block length. In other words, the advantage of BICM-ID over TCM for transmissions over AWGN channels is more significant for larger block lengths. The 8-state TTCM performance also improves, when using four iterations, as the block length is increased and, on the whole, TTCM is the best performer in this scenario.

Figure 13.25 shows the effects of the decoding complexity on the TCM, TTCM, BICM and BICM-ID schemes' performance in the context of an 8PSK scheme for transmissions over AWGN channels using a block length of 4000 information bits (2000 symbols). Again, the 64-state TCM, 64-state BICM, 8-state TTCM using four iterations and 16-state BICM-ID along with four iterations exhibit a similar decoding complexity. At a BER of 10^{-4}, TTCM requires about 0.6 dB lower SNR than BICM-ID, 1.6 dB less energy than TCM and 2.5 dB lower SNR than BICM. When the decoding complexity is reduced such that 8-state codes are used in the TCM, BICM and BICM-ID schemes, their corresponding performance becomes worse than that of the 64-state codes, as shown in Figure 13.25. In order to be able to compare the associated performance with that of 8-state BICM-ID using four iterations, 8-state TTCM along with two iterations is employed. Observe that due to the insufficient number of iterations, TTCM exhibits only marginal advantage over BICM-ID.

Figure 13.26 shows the performance of TCM, TTCM and BICM-ID invoking 16QAM for transmissions over AWGN channels using a block length of 6000 information bits (2000 symbols). Upon comparing 64-state TCM with 32-state BICM-ID using two iterations, we observed that BICM-ID outperforms TCM for E_b/N_0 values in excess of 6.8 dB. However, 8-state BICM-ID using an increased number of iterations, such as four or eight, outperforms the similar complexity 32-state BICM-ID scheme employing two iterations as well as 64-state TCM. An approximately 1.2 dB E_b/N_0 gain was obtained at a BER of 10^{-4} for 8-state BICM-ID using eight iterations over 64-state TCM at a similar decoding complexity. Comparing 8-state TTCM using two iterations and 8-state BICM-ID employing four iterations reveals that BICM-ID performs better for the E_b/N_0 range of 5.7 dB to 7 dB. When the number of iterations is increased to four for TTCM and to eight for BICM-ID, TTCM exhibits a better performance, as seen in Figure 13.26.

Owing to the associated SP, the intra-subset distance of TCM and TTCM increases as we traverse down the partition tree of Figure 13.7, for example. It was shown in [92] that we only need to encode $\tilde{m} = 2$ out

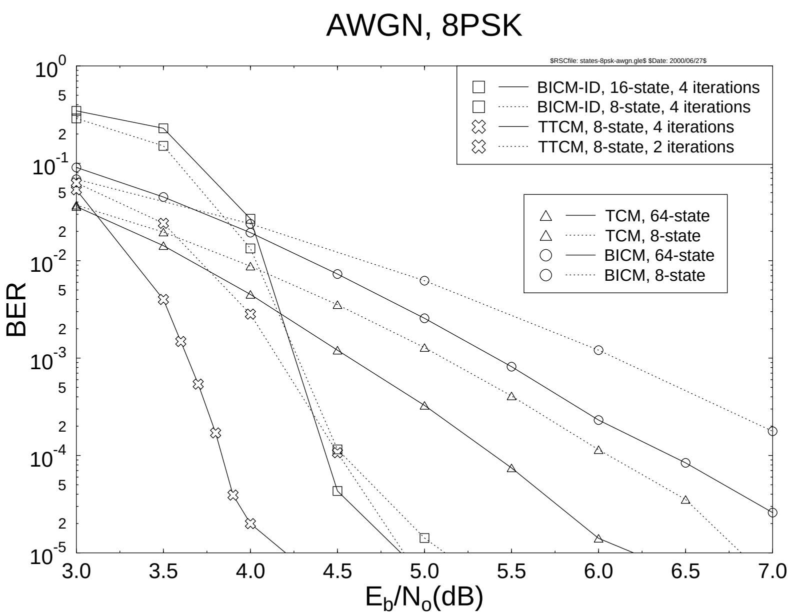


Figure 13.25: Effects of decoding complexity on the TCM, TTCM, BICM and BICM-ID schemes' performance in the context of an 8PSK scheme for transmissions over AWGN channels using a block length of 4000 information bits (2000 symbols).

of $\mathsf{m} = 5$ information bits in the 64QAM/TTCM to attain target BERs around 10^{-5} in AWGN channels. Hence in this scenario there are $2^{\mathsf{m}-\tilde{\mathsf{m}}} = 8$ parallel transitions due to the $\mathsf{m} - \tilde{\mathsf{m}} = 3$ uncoded information bits in the trellis of 64QAM/TTCM. Figure 13.27 illustrates the performance of TCM, TTCM, BICM and BICM-ID using 64QAM over AWGN channels. When using a block length of 10000 information bits (2000 symbols), 8-state TTCM invoking four iterations is the best candidate, followed by the similar complexity 8-state BICM-ID scheme employing eight iterations. Again, TCM performs better than BICM in AWGN channels. When a block length of 1250 information bits (250 symbols) was used, both TTCM and BICM-ID experienced a performance degradation. It is also seen in Figure 13.27 that BICM-ID performs close to TTCM, when a longer block length is used.

13.7.2.2.2 Performance over Uncorrelated Narrowband Rayleigh Fading Channels The uncorrelated Rayleigh fading channels implied using an infinite-length interleaver over narrowband Rayleigh fading channels. Figure 13.28 shows the performance of 64-state TCM, 64-state BICM, 8-state TTCM using four iterations and 16-state BICM-ID employing four iterations in the context of an 8PSK scheme communicating over uncorrelated narrowband Rayleigh fading channels using a block length of 4000 information bits (2000 symbols). These four coded modulation schemes have a similar complexity. As can be seen from Figure 13.28, TTCM performs best, followed by BICM-ID, BICM and TCM. At a BER of 10^{-4}, TTCM performs about 0.7 dB better in terms of the required E_b/N_0 value than BICM-ID, 2.3 dB better than BICM and 4.5 dB better than TCM. The error floor of TTCM [92] was lower than the associated EFF bound of BICM-ID. However, the BERs of TTCM and BICM-ID were identical at $E_b/N_0 = 7$ dB.

Figure 13.29 compares the performance of TCM, TTCM and BICM-ID invoking 16QAM for communicating over uncorrelated narrowband Rayleigh fading channels using a block length of 6000 information bits (2000 symbols). Observe that 32-state BICM-ID using two iterations outperforms 64-state TCM for E_b/N_0 in excess of 9.6 dB. At the same complexity, 8-state BICM-ID invoking eight iterations outperforms 64-state TCM beyond $E_b/N_0 = 8.2$ dB. Similarly to 8PSK, the coding gain of BICM-ID over TCM in the context of 16QAM is more significant over narrowband Rayleigh fading channels compared to AWGN

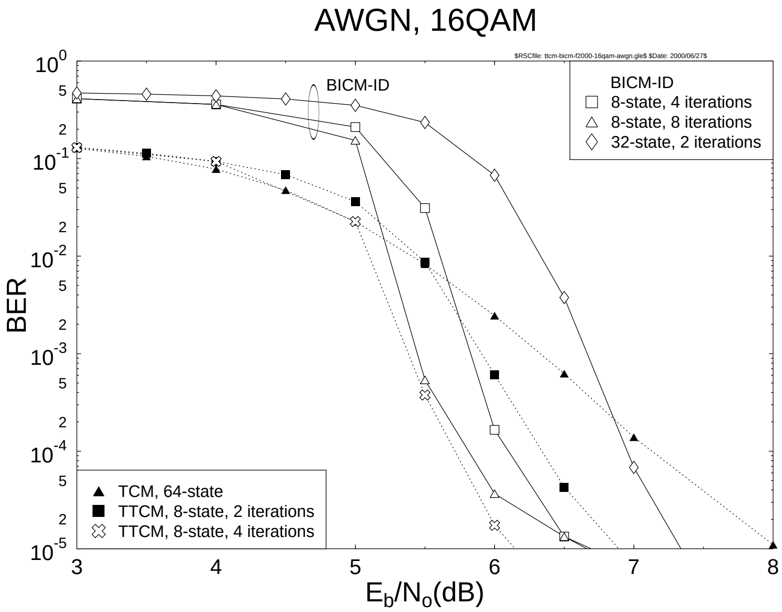


Figure 13.26: Performance comparison of TCM, TTCM and BICM-ID employing 16QAM for transmissions over AWGN channels using a block length of 6000 information bits (2000 symbols).

channels. Near E_b/N_0 of 11 dB the 8-state BICM-ID scheme approaches the EFF bound, hence 32-state BICM-ID using two iterations exhibits a better performance due to its lower EFF bound. Observe also that 8-state BICM-ID using four iterations outperforms 8-state TTCM employing two iterations in the range of $E_b/N_0 = 8.5$ dB to 12.1 dB. Increasing the number of iterations only marginally improves the performance of BICM-ID, but results in a significant gain for TTCM. The performance of 8-state TTCM using four iterations is better than that of 8-state BICM-ID along with eight iterations for E_b/N_0 values in excess of 9.6 dB.

Figure 13.30 illustrates the performance of TCM, TTCM, BICM and BICM-ID when invoking 64QAM for communicating over uncorrelated narrowband Rayleigh fading channels. Using a block length of 10000 information bits (2000 symbols), 64-state BICM performs better than 64-state TCM for E_b/N_0 values in excess of 15 dB. BICM-ID exhibits a lower error floor than TTCM in this scenario, since BICM-ID protects all the five information bits, while TTCM protects only two information bits of the six-bit 64QAM symbol. The three unprotected information bits of TCM and TTCM render these schemes less robust to the bursty error effects of the uncorrelated fading channel. If we use a TCM or TTCM code generator that encodes all the five information bits, a better performance is expected. Reducing the block length from 2000 symbols to 250 symbols resulted in a small performance degradation for TTCM, but yielded a significant degradation for BICM-ID.

13.7.2.2.3 Coding Gain versus Complexity and Interleaver Block Length In this section, we will investigate the coding gain (G) of the coded modulation schemes utilising an 8PSK scheme versus the Decoding Complexity (DC) and the Interleaver Block Length (IL) at a BER of 10^{-4}. The coding gain G is measured by comparing to the uncoded 4PSK scheme, which exhibits a BER of 10^{-4} at $E_b/N_0 = 8.35$ dB and $E_b/N_0 = 35$ dB for transmissions over AWGN channels and uncorrelated narrowband Rayleigh fading channels, respectively. Again, the DC is measured using the associated number of decoding states and the notations S and t represent the number of decoding states and the number of decoding iterations, respectively. Hence, the relative complexity of TCM, BICM, TTCM and BICM-ID is given by S, S, $2 \times t \times S$ and $t \times S$, respectively. The IL is measured in terms of the number of information bits in the interleaver.

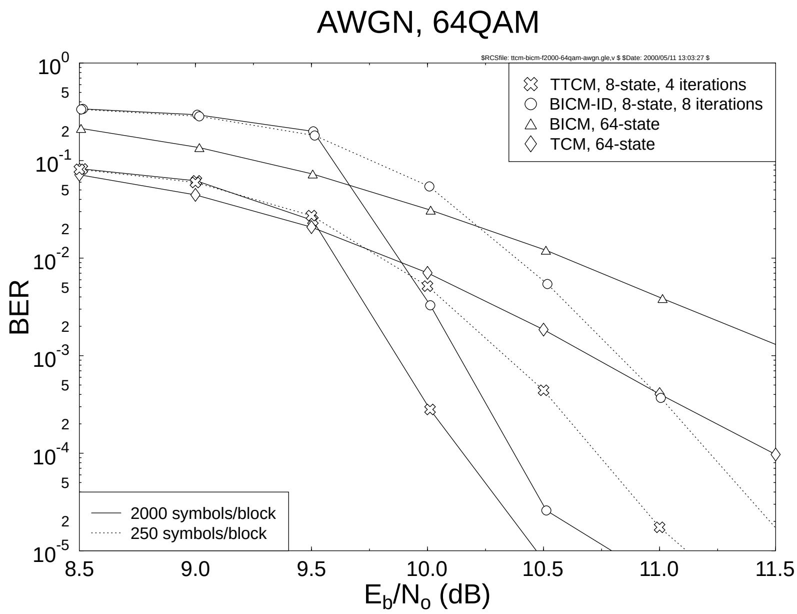


Figure 13.27: Performance comparison of TCM, TTCM, BICM and BICM-ID using 64QAM over AWGN channels.

Figure 13.31 portrays the coding gain G versus DC plot of the coded modulation schemes for 8PSK transmissions over (a) AWGN channels and (b) uncorrelated narrowband Rayleigh fading channels, using an IL of 4000 information bits (2000 symbols). At a DC as low as 8, the non-iterative TCM scheme exhibits the highest coding gain G for transmissions over AWGN channels, as seen in Figure 13.31(a). By contrast, the BICM scheme exhibits the highest coding gain G for transmissions over uncorrelated narrowband Rayleigh fading channels, as seen in Figure 13.31(b). However, for a DC higher than 16, the iterative TTCM and BICM-ID schemes exhibit higher coding gains than their non-iterative counterparts for transmission over both channels.

For the iterative schemes different combinations of t and S may yield different performances at the same DC. For example, the coding gain G of BICM-ID in conjunction with $t \cdot S = 8 \times 8$ is better than that of $t \cdot S = 4 \times 16$ at DC=64 for transmissions over AWGN channels, as seen in Figure 13.31(a), since BICM-ID invoking a constituent code associated with $S = 16$ has not reached its optimum performance at iteration $t = 4$. However, the coding gain G of BICM-ID in conjunction with $t \cdot S = 4 \times 16$ is better than that of $t \cdot S = 8 \times 8$ at DC=64, when communicating over uncorrelated narrowband Rayleigh fading channels, as seen in Figure 13.31(b). This is because BICM-ID invoking a constituent code associated with $S = 8$ has reached its EFF bound at iteration $t = 4$, while BICM-ID invoking a constituent code associated with $S = 16$ has not reached its EFF bound, because the EFF bound for code associated with $S = 16$ is lower than that of a code associated with $S = 18$. In general, the coding gain G of TTCM is the highest for DC values in excess of 32 for transmissions over both channels.

Figure 13.32 portrays the coding gain G versus IL plot of the coded modulation schemes for 8PSK transmissions over (a) AWGN channels and (b) uncorrelated narrowband Rayleigh fading channels in conjunction with a DC of 64 both with and without code termination. We can observe in Figure 13.32(a) that IL affects the performance of the schemes using no code termination, since the shorter the IL, the higher the probability for the decoding trellis to end at a wrong state. For transmissions over AWGN channels and upon using code-terminated schemes, only the performance of the BICM-ID scheme is affected by the IL, since the performance of the scheme communicating over AWGN channels depends on the FED, while the high FED of BICM-ID depends on the reliability of the feedback values. Therefore, when the IL is

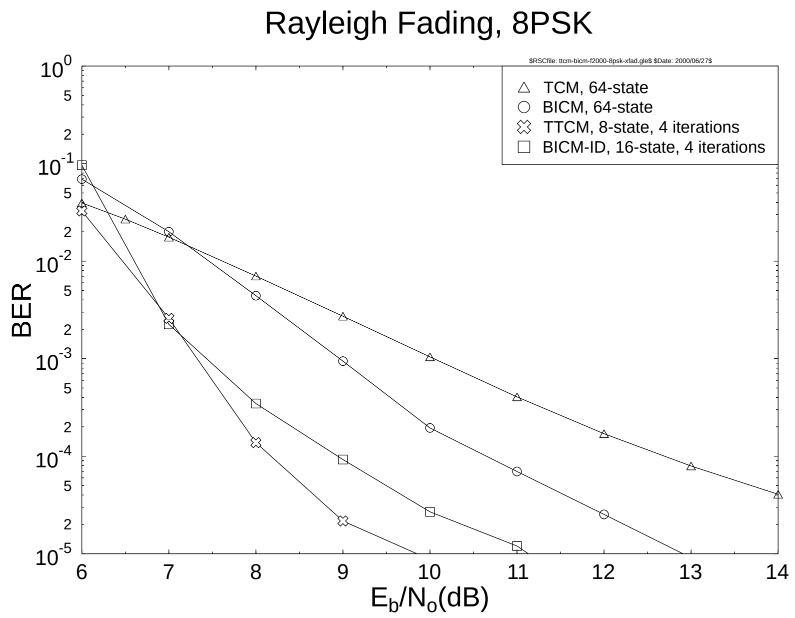


Figure 13.28: Performance comparison of TCM, TTCM, BICM and BICM-ID for 8PSK transmissions over uncorrelated Rayleigh fading channels using a block length of 4000 information bits (2000 symbols).

short, BICM-ID suffers from a performance degradation. However, the other schemes are not affected by the IL when communicating over AWGN channels, as seen in Figure 13.32(a), since there are no bursty channel errors to be dispersed by the interleaver and hence there is no advantage in utilising a long IL. To elaborate a little further, as seen in Figure 13.32 for transmissions over uncorrelated narrowband Rayleigh fading channels using code-terminated schemes, the IL does not significantly affect the performance of the schemes, since the error events are uncorrelated in the uncorrelated Rayleigh fading scenario. These results constitute the upper bound performance achievable when an infinitely long interleaver is utilised for rendering the error events uncorrelated.

Figure 13.33 portrays the coding gain G versus IL plot of the coded modulation schemes for 8PSK transmissions over correlated narrowband Rayleigh fading channels, in conjunction with a decoding complexity of 64, when applying code termination. The normalised Doppler frequency of the channel is 3.25×10^{-5}, which corresponds to a Baud rate of 2.6 MBaud, a carrier frequency of 1.9 GHz and a vehicular speed of 30 mph. This is a slow fading channel and hence the fading envelope is highly correlated. It is demosntrated by Figure 13.33 that the coding gain G of all coded modulation schemes improves as the IL increases. This is because the MAP decoder is unable to perform at its best when the channel errors occur in bursts. However, the performance improves when the error bursts are dispersed by the employment of a long interleaver. In general, TTCM is the best performer for a variety of IL values. However, BICM-ID is the worst performer for an IL of 4000 bits, while performing similarly to TTCM for long IL values.

On one hand, TCM performs better than BICM for short IL values, which follows the performance trends observed for transmissions over AWGN channels, as shown in Figure 13.32(a). This is because slowly fading channels are highly correlated and hence they behave as near-Gaussian channels, where TCM is at its best, since TCM was designed for Gaussian channels. By contrast, although BICM was designed for fading channels, when the channel-induced error bursts are inadequately dispersed owing to the employment of a short IL, the performance of BICM suffers. In other words, when communicating over slowly fading channels, extremely long interleavers may be necessary for over-bridging the associated long fades and for facilitating the dispersion of bursty transmission errors, which is a prerequisite for the efficient

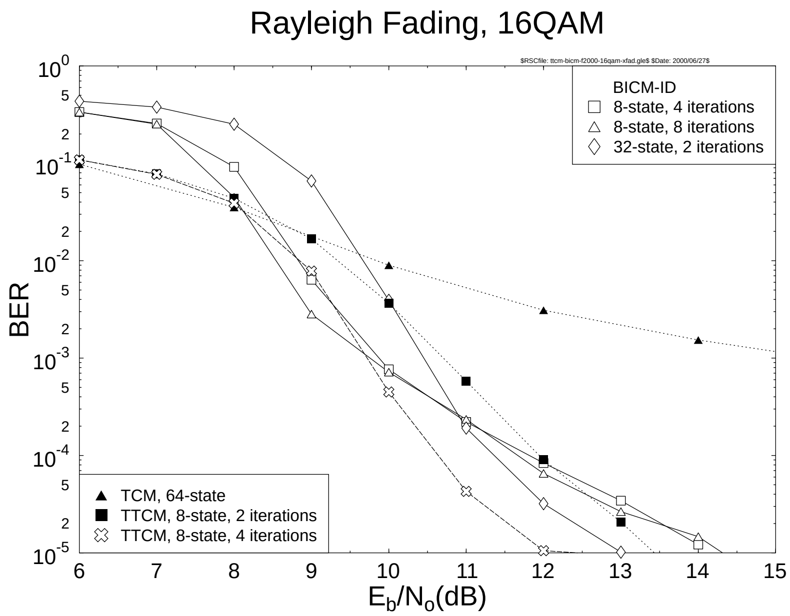


Figure 13.29: Performance comparison of TCM, TTCM and BICM-ID for 16QAM transmissions over uncorrelated narrowband Rayleigh fading channels transmitting 2000 symbols/block (6000 information bits/block).

operation of channel codecs.

On the other hand, BICM performs better than TCM for long IL values, which is reminiscent of the performance trends observed when communicating over uncorrelated Rayleigh fading channels, as evidenced by Figure 13.32(b). This is justified, since the correlation of the fading channel is broken when a long IL is employed for dispersing the error bursts.

13.7.2.3 Conclusion

In conclusion, at a given complexity TCM performs better than BICM in AWGN channels, but worse in uncorrelated narrowband Rayleigh fading channels. However, BICM-ID using soft-decision feedback outperforms TCM and BICM for transmissions over both AWGN and uncorrelated narrowband Rayleigh fading channels at the same DC. TTCM has shown superior performance over the other coded modulation schemes studied, but exhibited a higher error floor for the 64QAM scheme due to the presence of uncoded information bits for transmissions over uncorrelated narrowband Rayleigh fading channels. Comparing the coding gain against the DC, the iterative decoding schemes of TTCM and BICM-ID are capable of providing a high coding gain even in conjunction with a constituent code exhibiting a short memory length, although only at the cost of a sufficiently high number of decoding iterations, which may imply a relatively high decoding complexity. Comparing the achievable coding gain against the IL, TTCM is the best performer for a variety of ILs, while the performance of BICM-ID is highly dependent on the IL for transmissions over both AWGN and Rayleigh fading channels.

13.7.3 Coded Modulation in Wideband Channels

In this section we will consider the performance of the various coded modulation schemes in the context of practical dispersive channels.

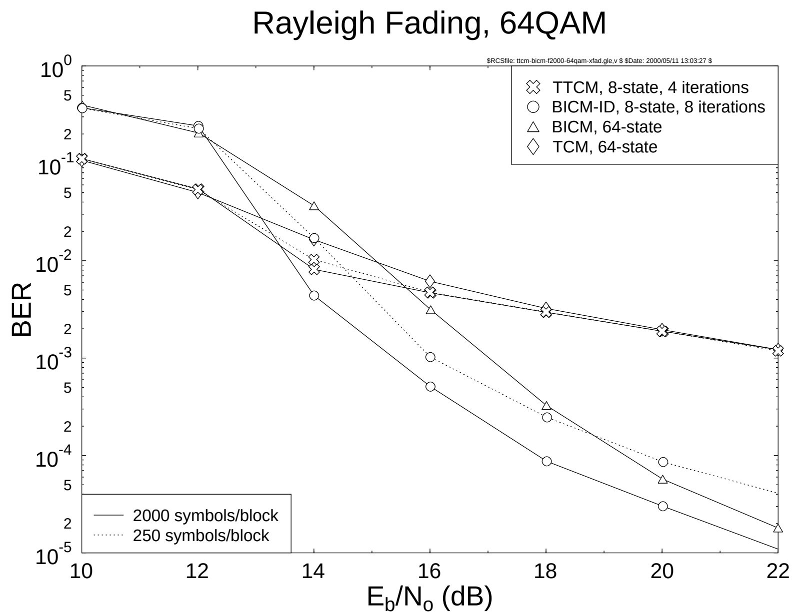


Figure 13.30: Performance comparison of TCM, TTCM, BICM and BICM-ID using 64QAM for transmissions over uncorrelated narrowband Rayleigh fading channels.

13.7.3.1 Inter-symbol Interference

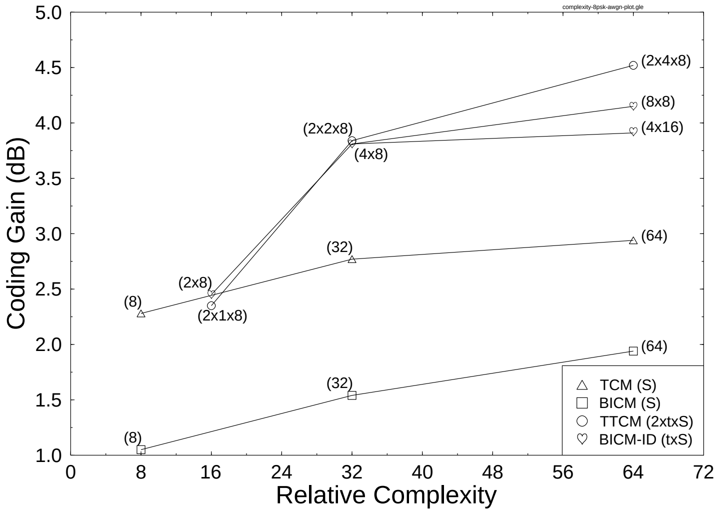
The mobile radio channels [49] can be typically characterised by band-limited linear filters. If the modulation bandwidth exceeds the coherence bandwidth of the radio channel, Inter-Symbol Interference (ISI) occurs and the modulation pulses are spread or dispersed in the time domain. The ISI, inflicted by band-limited frequency, selective time-dispersive channels, distorts the transmitted signals. At the receiver, the linearly distorted signal has to be equalised in order to recover the information.

The linearly distorted instantaneous signal received over the dispersive channel can be visualised as the superposition of the channel's response due to several information symbols in the past and in the future. Figure 13.34 shows the Channel's Impulse Response (CIR) exhibiting three distinct parts. The main tap h_2 possesses the highest relative amplitude. The taps before the main tap, namely h_0 and h_1, are referred to as pre-cursors, whereas those following the main tap, namely h_3 and h_4, are referred to as post-cursors.

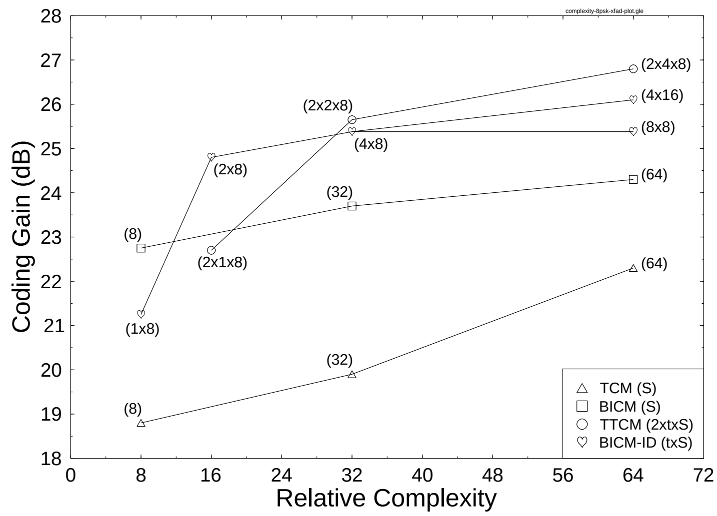
The energy of the wanted signal is received mainly over the path described by the main channel tap. However, some of the received energy is contributed by the convolution of the pre-cursors with future symbols and the convolution of the post-cursor with past symbols, which are termed pre-cursor ISI and post-cursor ISI, respectively. Thus the received signal is constituted by the superposition of the wanted signal, pre-cursor ISI and post-cursor ISI.

13.7.3.2 Decision Feedback Equaliser

Channel equalisers that are utilised for compensating the effects of ISI can be classified structurally as linear equalisers or DFEs. They can be distinguished also on the basis of the criterion used for optimising their coefficients. When applying the Minimum Mean Square Error (MMSE) criterion, the equaliser is optimised such that the mean squared error between the distorted signal and the actual transmitted signal is minimised. For time varying dispersive channels, a range of adaptive algorithms can be invoked for updating the equaliser coefficients and for tracking the channel variations [193].

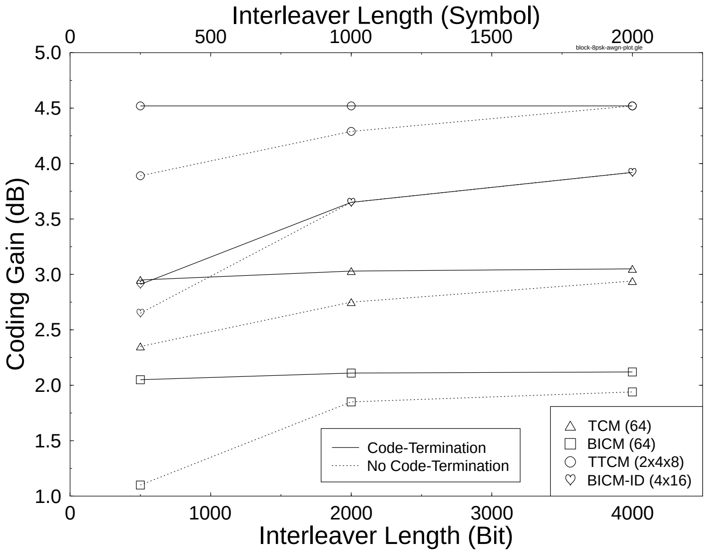


(a) AWGN channels

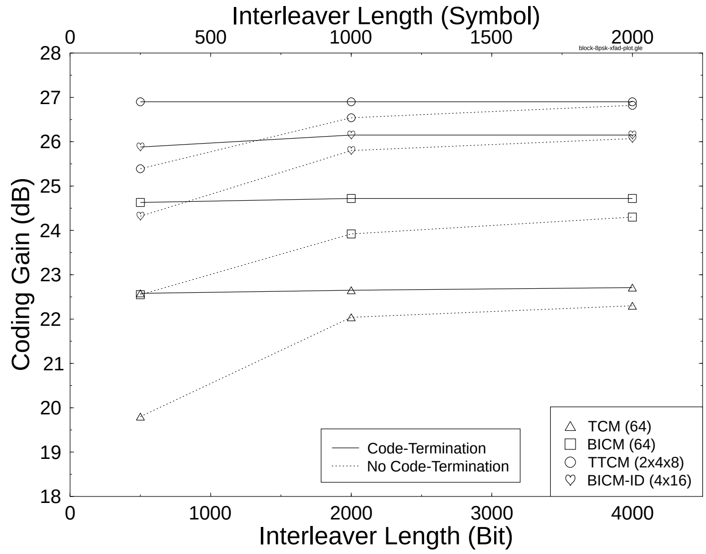


(b) uncorrelated narrowband Rayleigh fading channels

Figure 13.31: Coding gain at a BER of 10^{-4} over the uncoded 4PSK scheme, against the decoding complexity of TCM, TTCM, BICM and BICM-ID for 8PSK transmissions over (a) AWGN channels and (b) uncorrelated narrowband Rayleigh fading channels, using an interleaver block length of 4000 information bits (2000 symbols). The notations S and t represent the number of decoding states and the number of decoding iterations, respectively.



(a) AWGN channels



(b) uncorrelated narrowband Rayleigh fading channels

Figure 13.32: Coding gain at a BER of 10^{-4} over the uncoded 4PSK scheme, against the IL of TCM, TTCM, BICM and BICM-ID for 8PSK transmissions over (a) AWGN channels and (b) uncorrelated narrowband Rayleigh fading channels, invoking a DC of 64 applying code termination or no code termination.

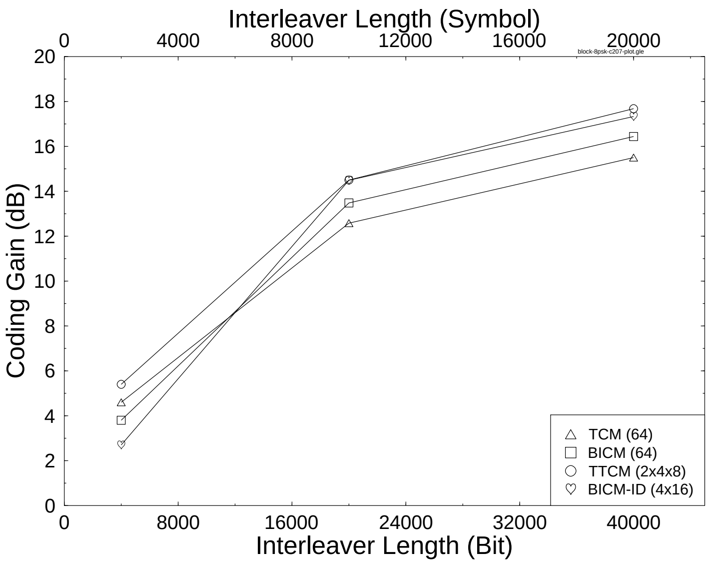


Figure 13.33: Coding gain at a BER of 10^{-4} over the uncoded 4PSK scheme, against the IL of TCM, TTCM, BICM and BICM-ID for 8PSK transmissions over correlated narrowband Rayleigh fading channels, invoking a DC of 64 applying code termination. The normalised Doppler frequency of the channel is 3.25×10^{-5}, which corresponds to a Baud rate of 2.6 MBaud, a carrier frequency of 1.9 GHz and a vehicular speed of 30 mph.

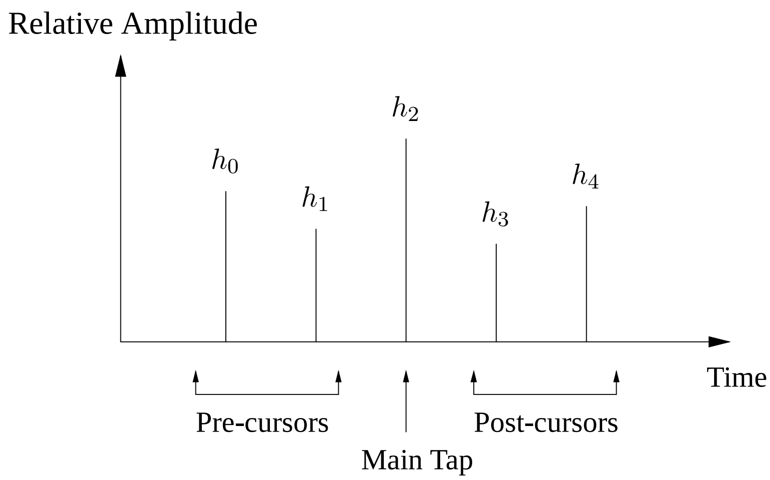


Figure 13.34: Channel Impulse Response (CIR) having pre-cursors, the main tap and post-cursors.

13.7.3.2.1 Decision Feedback Equaliser Principle The simple Zero Forcing Equaliser (ZFE) [314] forces all the impulse response contributions of the concatenated system constituted by the channel and the equaliser to zero at the signalling instants nT for $n \neq 0$, where T is the signalling interval duration. The ZFE provides gain in the frequency domain at frequencies where the channel's transfer function experiences attenuation and vice versa. However, both the signal and the noise are enhanced simultaneously and hence the ZFE is ineffective owing to the associated noise enhancement effects. Furthermore, no finite-gain ZFE can be designed for channels that exhibit spectral nulls [193, 288].

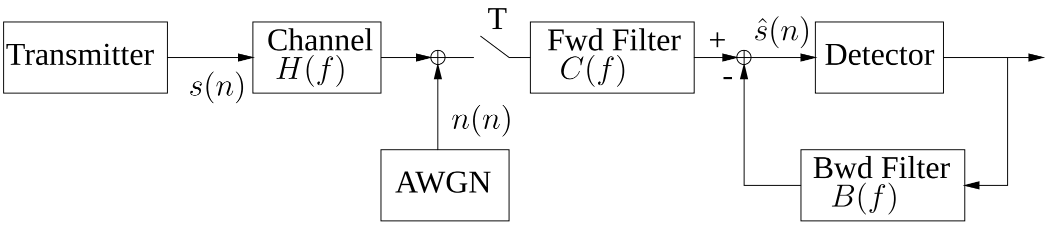


Figure 13.35: Schematic of the transmission system portraying the oriented feedforward (Fwd) and backward (Bwd) filter of the DFE, where $C(f)$ and $B(f)$ are the corresponding frequency-domain transfer functions, respectively.

Linear MMSE equalisers [314] are designed for mitigating both the pre-cursor ISI and the post-cursor ISI, as defined in Section 13.7.3.1. The MMSE equaliser is more intelligent than the ZFE, since it jointly minimises the effects of both the ISI and noise. Although the linear MMSE equaliser approaches the same performance as the ZFE at high SNRs, an MMSE solution does exists for all channels, including those that exhibit spectral nulls.

The idea behind the DFE [193, 288, 314] is that once an information symbol has been detected and decided upon, the ISI that these detected symbols inflicted on future symbols can be estimated and the corresponding ISI can be cancelled before the detection of subsequent symbols.

The DFE employs a feedforward filter and a backward-oriented filter for combating the effects of dispersive channels. Figure 13.35 shows the general block diagram of the transmission system employing a DFE. The forward-oriented filter partially eliminates the ISI introduced by the dispersive channel. The feedback filter, in the absence of decision errors, is fed with the error-free transmitted signal in order to further reduce the ISI.

The feedback filter, denoted as the Bwd Filter in Figure 13.35, receives the detected symbol. Its output is then subtracted from the estimates generated by the forward filter, denoted as the Fwd Filter, in order to produce the detector's input. Since the feedback filter uses the ISI-free signal as its input, the feedback loop mitigates the ISI without introducing enhanced noise into the system. The drawback of the DFE is that when wrong decisions are fed back into the feedback loop, error propagation is inflicted and the BER performance of the equaliser is degraded.

The detailed DFE structure is shown in Figure 13.36. The feedforward filter is constituted by the coefficients or taps labelled as $C_0 \rightarrow C_{N_f-1}$, where N_f is the number of taps in the feedforward filter, as shown in the figure. The causal feedback filter is constituted by N_b feedback taps, denoted as $b_1 \rightarrow b_{N_b}$. Note that the feedforward filter contains only the present input signal r_k, and future input signals $r_{k+1} \ldots r_{k+(N_f+1)}$, which implies that no latency is inflicted. Therefore, the feedforward filter eliminates only the pre-cursor ISI, but not the post-cursor ISI. By contrast, the feedback filter mitigates the ISI caused by the past data symbols, i.e. post-cursor ISI. Since the feedforward filter only eliminates the pre-cursor ISI, the noise enhancement effects are less problematic in DFEs compared to the linear MMSE equaliser.

Here, the MMSE criterion [314] is used for deriving the optimum coefficients of the feedforward section of the DFE. The Mean Square Error (MSE) between the transmitted signal, s_k, and its estimate, $\hat{s}_k$, at the equaliser's output is formulated as:

$$MSE = E[|s_k - \hat{s}|^2], \tag{13.43}$$

where $E[|s_k - \hat{s}|^2]$ denotes the expected value of $|s_k - \hat{s}|^2$. In order to minimise the MSE, the orthogonality principal [315] is applied, stating that the residual error of the equaliser, $e_k = s_k - \hat{s}_k$, is orthogonal to the input signal of the equaliser, r_k, when the equaliser taps are optimal, yielding:

$$E[e_k r^*_{k+l}] = 0, \tag{13.44}$$

where the superscript * denotes conjugation. Following Cheung's approach [193, 316], the optimum coef-

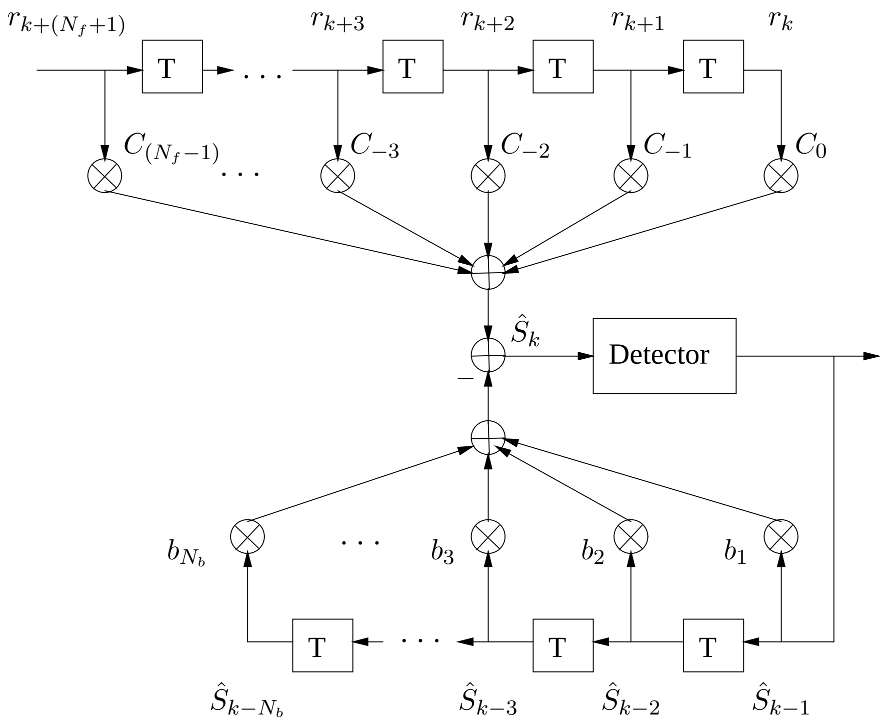


Figure 13.36: Structure of the DFE where r_k and $\hat{S}_k$ denote the received signal and detected symbol, respectively, while C_m, b_q represent the coefficient taps of the forward- and backward-oriented filters, respectively.

ficient of the feedforward section can be derived from the following set of N_f equations:

$$\sum_{m=0}^{N_f-1} C_m \left[\sum_{v=0}^{l} h_v^* h_{v+m-l} \sigma_S^2 + N_0 \delta_{m-l} \right] = h_l^* \sigma_S^2, \; l = 0 \; \ldots N_f - 1, \tag{13.45}$$

where σ_s^2 and $N_0/2$ are the signal and noise variance, respectively, while h^* denotes the complex conjugate of the CIR and δ is the delta function. By solving these N_f simultaneous equations, the equaliser coefficients, C_m, can be obtained. For the feedback filter the following set of N_b equations were used, in order to derive the optimum feedback coefficient, b_q [316]:

$$b_q = \sum_{m=0}^{N_f-1} C_m h_{m+q}, \; q = 1 \; \ldots N_b. \tag{13.46}$$

13.7.3.2.2 Equalizer Signal To Noise Ratio Loss The equaliser's performance can be measured in terms of the equaliser's SNR loss, BER performance and MSE [288]. Here the SNR loss is considered, since this parameter will be used in next section.

The SNR loss of the equaliser was defined by Cheung [316] as:

$$SNR_{loss} = SNR_{input} - SNR_{output}, \tag{13.47}$$

where SNR_{input} is the SNR measured at the equaliser's input, given by:

$$SNR_{input} = \frac{\sigma_s^2}{2\sigma_N^2}, \tag{13.48}$$

with σ_s^2 being the average received signal power, assuming wide sense stationary conditions, and σ_N^2 is the variance of the AWGN.

The equaliser's output contains the wanted signal, the effective Gaussian noise, the residual ISI and the ISI caused by the past data symbols. In order to simplify the calculation of SNR_{output}, we assume that the SNR is high and hence we consider the low-BER range, where effectively correct bits are fed back to the DFE's feedback filter. Thus the post-cursor ISI is completely eliminated from the equaliser's output. Hence SNR_{output} is given by [316]:

$$SNR_{output} = \frac{\text{Wanted Signal Power}}{\text{Residual ISI Power } + \text{ Effective Noise Power}}, \tag{13.49}$$

where the residual ISI is assumed to be an extra noise source that possessed a Gaussian distribution. Therefore, we have:

$$\text{Wanted Signal Power} = E\left[|s_k \sum_{m=0}^{N_f-1} C_m h_m|^2 \right], \tag{13.50}$$

$$\text{Effective Noise Power} = N_0 \sum_{i=0}^{N_f-1} |C_i|^2, \tag{13.51}$$

and:

$$\text{Residual ISI Power} = \sum_{q=-(N_f-1)}^{-1} E\left[|f_q s_{k-q}|^2 \right], \tag{13.52}$$

where $f_q = \sum_{m=0}^{N_f-1} C_m h_{m+q}$ and the remaining notations are accrued from Figure 13.36. By substituting Equations 13.50, 13.51 and 13.52 into 13.49, the SNR_{output} can be written as [288]:

$$SNR_{output} = \frac{E\left[|s_k \sum_{m=0}^{N_f-1} C_m h_m|^2 \right]}{\sum_{q=-(N_f-1)}^{-1} E\left[|f_q s_{k-q}|^2 \right] \; + \; N_0 \sum_{i=0}^{N_f-1} |C_i|^2}. \tag{13.53}$$

Following this rudimentary introduction to channel equalisation, we focus our attention on quantifying the performance of various wideband coded modulation schemes, referring the reader to [193] for an in-depth discourse on channel equalisation.

13.7.3.3 Decision Feedback Equalizer Aided Adaptive Coded Modulation

In this section, DFE-aided wideband Burst-by-Burst (BbB) adaptive TCM, TTCM, BICM and BICM-ID schemes are proposed and characterised in performance terms, when communicating over the COST207 Typical Urban (TU) wideband fading channel. These schemes are evaluated using a practical near-instantaneous modem mode switching regime. **System I** represents schemes without channel interleaving, while **System II** invokes channel interleaving over four transmission bursts. **System I** exhibited a factor four delay in lower overall modem mode signalling, and hence it was capable of more prompt modem mode reconfiguration. By contrast, **System II** was less agile in terms of modem mode reconfiguration, but benefited from a longer interleaver delay. We will show in Section 13.7.3.3.4 that a substantially improved Bit Per Symbol (BPS) and BER performance was achieved by **System II** in comparison to **System I**. We will also show that BbB adaptive TTCM was found to perform better than the BbB adaptive TCM in **System II** at a similar DC, when aiming for a target BER of below 0.01%.

13.7.3.3.1 Introduction In general fixed-mode transceivers fail to adequately counteract the time varying nature of the mobile radio channel and hence typically result in bursts of transmission errors. By contrast, in BbB adaptive schemes [209] a higher-order modulation mode is employed when the instantaneous estimated channel quality is high in order to increase the number of BPS transmitted and, conversely, a more robust lower-order modulation mode is employed when the instantaneous channel quality is low, in order to improve the mean BER performance. Uncoded adaptive schemes [209, 211, 223, 224, 317, 318] and coded adaptive schemes [218, 319, 320] have been investigated for transmissions over narrowband fading channels. Finally, a turbo-coded wideband adaptive scheme assisted by a DFE was investigated in [321].

In our practical approach the local transmitter is informed about the channel quality estimate generated by the remote receiver upon receiving the transmission burst of the remote transmitter. In other words, the modem mode required by the remote receiver for maintaining its target integrity is superimposed on the transmission burst emitted by the remote transmitter. Hence a delay of one transmission burst duration is incurred. In the literature, adaptive coding designed for time varying channels using outdated fading estimates has been investigated for example in [322].

Over wideband fading channels the DFE employed will eliminate most of the ISI. Consequently, the MSE at the output of the DFE can be calculated and used as the metric invoked for switching the modulation modes [223]. This ensures that the performance is optimised by employing equalization and BbB adaptive TCM/TTCM jointly, in order to combat both the signal power fluctuations and the time variant ISI of the wideband channel.

In Section 13.7.3.3.2, the system's schematic is outlined. In Section 13.7.3.3.3, the performance of various fixed-mode TCM and TTCM schemes is evaluated, while Section 13.7.3.3.4 contains the detailed characterisation of the BbB adaptive TCM/TTCM schemes in the context of the non-interleaved **System I** and interleaved **System II**. In Section 13.7.3.3.5 we compare the performance of the proposed schemes with that of other adaptive coded modulation schemes, such as BICM and BICM-ID. Finally, we will conclude with our findings in Section 13.7.3.3.6.

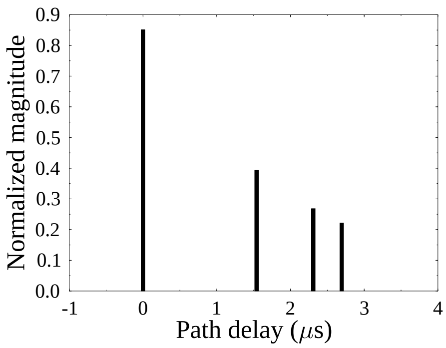


Figure 13.37: The impulse response of a COST207 Typical Urban (TU) channel [264].

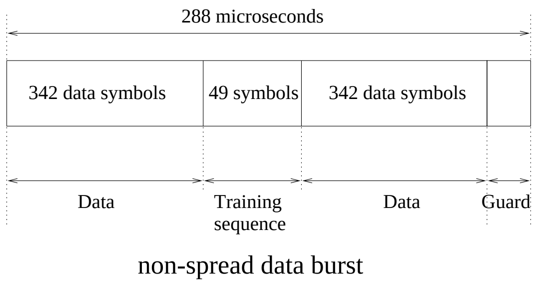


Figure 13.38: Transmission burst structure of the FMA1 non-spread data as specified in the FRAMES proposal [239].

13.7.3.3.2 System Overview The multi-path channel model is characterised by its discretised symbol-spaced COST207 Typical Urban (TU) CIR [264], as shown in Figure 13.37. Each path is faded independently according to a Rayleigh distribution and the corresponding normalised Doppler frequency is 3.25×10^{-5}, the system's Baud rate is 2.6 MBaud, the carrier frequency is 1.9 GHz and the vehicular speed is 30 mph. The DFE incorporated 35 feed forward taps and 7 feedback taps and the transmission burst structure used is shown in Figure 13.38. When considering a Time Division Multiple Access (TDMA)/Time Division Duplex (TDD) system providing 16 slots per 4.615 ms TDMA frame, the transmission burst duration is 288 μs, as specified in the Pan-European FRAMES proposal [239].

The following assumptions are stipulated. First, we assume that the equaliser is capable of estimating the CIR perfectly with the aid of the equaliser training sequence of Figure 13.38. Second, the CIR is time-invariant for the duration of a transmission burst, but varies from burst to burst according to the Doppler frequency, which corresponds to assuming that the CIR is slowly varying. We refer to this scenario as encountering burst-invariant fading. The error propagation of the DFE will degrade the estimated performance, but the effect of error propagation is left for further study. At the receiver, the CIR is estimated, and is then used for calculating the DFE coefficients [193]. Subsequently, the DFE is used for equalising the ISI-corrupted received signal. In addition, both the CIR estimate and the DFE feedforward coefficients are utilised for computing the SNR at the output of the DFE. More specifically, by assuming that the residual ISI is near-Gaussian distributed and that the probability of decision feedback errors is negligible, the SNR at the output of the DFE, γ_{dfe}, is calculated as [316]:

$$\begin{aligned} \gamma_{dfe} &= \frac{\text{Wanted Signal Power}}{\text{Residual ISI Power + Effective Noise Power}}. \\ &= \frac{E\left[|s_k \sum_{m=0}^{N_f} C_m h_m|^2\right]}{\sum_{q=-(N_f-1)}^{-1} E\left[|\sum_{m=0}^{N_f-1} C_m h_{m+q} s_{k-q}|^2\right] + N_0 \sum_{m=0}^{N_f} |C_m|^2}, \end{aligned} \tag{13.54}$$

where C_m and h_m denote the DFE's feedforward coefficients and the CIR, respectively. The transmitted signal is represented by s_k and N_0 denotes the noise spectral density. Finally, the number of DFE feedforward coefficients is denoted by N_f.

The equaliser's SNR, γ_{dfe}, in Equation 13.54, is then compared against a set of adaptive modem mode switching thresholds f_n, and subsequently the appropriate modulation mode is selected [223, 323]. The modem mode required by the remote receiver for maintaining its target integrity is then fed back to the local transmitter. The modulation modes that are utilised in this scheme are 4QAM, 8PSK, 16QAM and 64QAM [193].

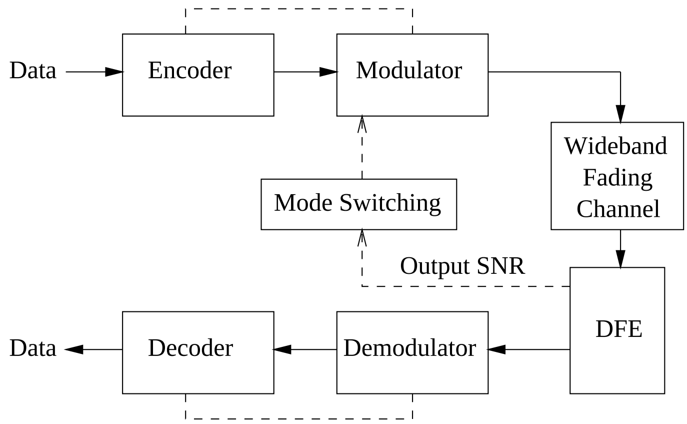


Figure 13.39: **System I** employing no channel interleaver. The equaliser's output SNR is used for selecting a suitable modulation mode, which is fed back to the transmitter on a burst-by-burst basis.

The simplified block diagram of the BbB adaptive TCM/TTCM **System I** is shown in Figure 13.39, where no channel interleaving is used. Transmitter A extracts the modulation mode required by receiver B from the reverse-link transmission burst in order to adjust the adaptive TCM/TTCM mode suitable for the currently experienced instantaneous channel quality. This incurs one TDMA/TDD frame delay between estimating the actual channel condition at receiver B and the selected modulation mode of transmitter A. Better channel quality prediction can be achieved using the techniques proposed in [324]. We invoke four encoders, each adding one parity bit to each information symbol, yielding the coding rate of $1/2$ in conjunction with the TCM/TTCM mode of 4QAM, $2/3$ for 8PSK, $3/4$ for 16QAM and $5/6$ for 64QAM.

The design of TCM schemes contrived for fading channels relies on the time and space diversity provided by the associated channel coder [291, 304]. Diversity may be achieved by repetition coding, which reduces the effective data rate, spacedtime- coded multiple transmitter/receiver structures [79], which increases cost and complexity, or by simple interleaving, which induces latency. In [325] adaptive TCM schemes were designed for narrowband fading channels utilising repetition-based transmissions during deep fades along with ideal channel interleavers and assuming zero delay for the feedback of the channel quality information.

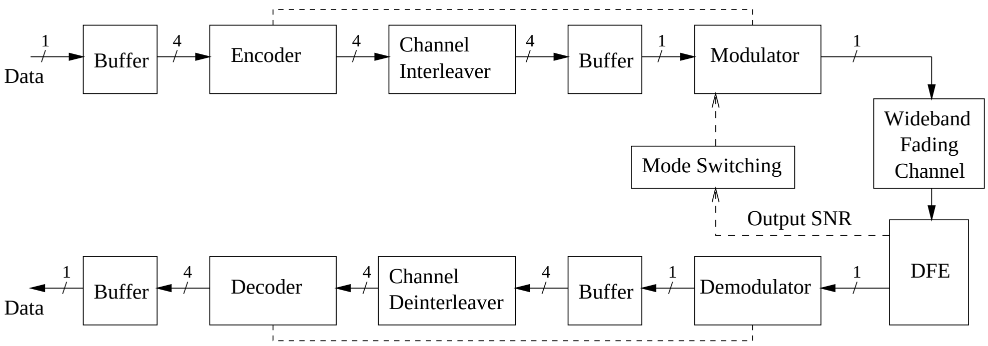


Figure 13.40: **System II** employing a channel interleaver length of four TDMA/TDD bursts. Data are entered into the input buffer on a burst-by-burst basis and the modulator modulates coded data read from the output buffer for transmission on a burst-by-burst basis. The encoder and channel interleaver as well as the decoder and channel de-interleaver operate on a four-burst basis. The equaliser's output SNR during the fourth burst is used for selecting a suitable modulation mode and fed back to the transmitter on the reverse-link burst.

Figure 13.40 shows the block diagram of **System II**, where symbol-based channel interleaving over four transmission bursts is utilised, in order to disperse the bursty symbol errors. Hence, the coded modulation module assembles four bursts using an identical modulation mode, so that they can be interleaved using the symbol-by-symbol random channel interleaver without the need of adding dummy bits. Then, these four-burst TCM/TTCM packets are transmitted to the receiver. Once the receiver has received the fourth burst, the equaliser's output SNR for this most recent burst is used for choosing a suitable modulation mode. The selected modulation mode is fed back to the transmitter on the reverse-link burst. Upon receiving the modulation mode required by receiver B (after one TDMA frame delay), the coded modulation module assembles four bursts of data from the input buffer for coding and interleaving, which are then stored in the output buffer ready for the next four bursts' transmission. Thus the first transmission burst exhibits one TDMA/TDD frame delay and the fourth transmission burst exhibits four frame delay which is the worst-case scenario.

Soft-decision trellis decoding utilising the Log-MAP algorithm [52] of Section 3.3.5 was invoked for TCM/TTCM decoding. The Log-MAP algorithm is a numerically stable version of the MAP algorithm operating in the log-domain, in order to reduce its complexity and to mitigate the numerical problems associated with the MAP algorithm [11]. As stated in Section 13.2, the TCM scheme invokes Ungerböck's codes [290], while the TTCM scheme invokes Robertson's codes [92]. A component TCM code memory of 3 was used for the TTCM scheme. The number of turbo iterations for TTCM was fixed to four and hence it exhibited a similar DC to the TCM code memory of 6. In the next section we present simulation results for our fixed-mode transmissions.

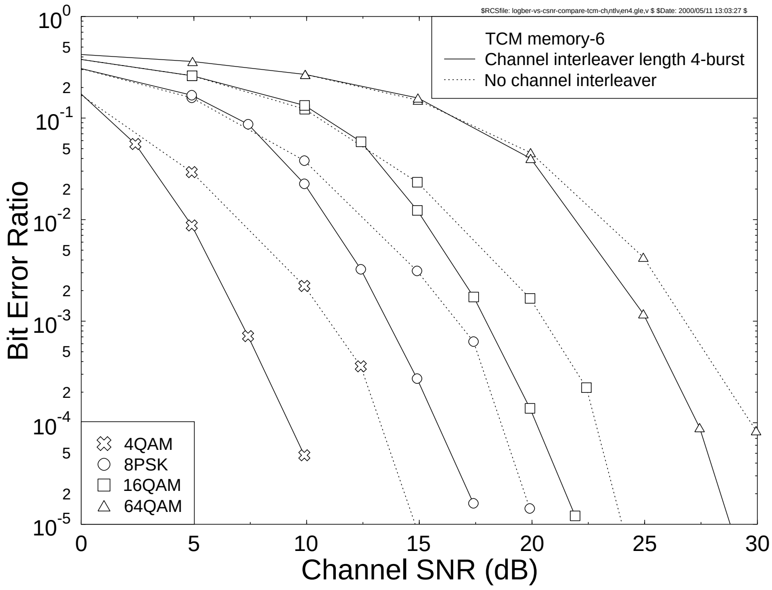


Figure 13.41: TCM performance of each individual modulation mode over the Rayleigh fading COST207 TU channel of Figure 13.37. A TCM code memory of 6 was used, since it had a similar decoding complexity to TTCM in conjunction with four iterations using a component TCM code memory of 3.

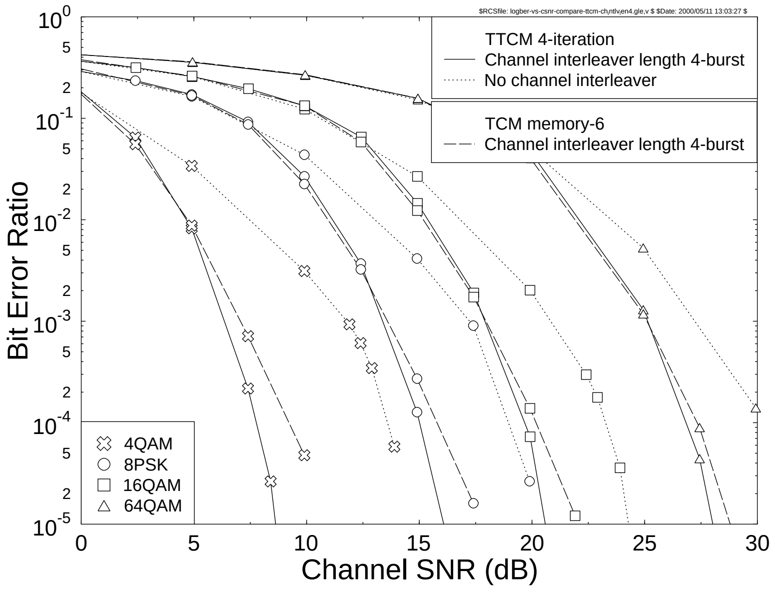


Figure 13.42: TTCM performance of each individual modulation mode over the Rayleigh fading COST207 TU channel of Figure 13.37. A component TCM code memory of 3 was used and the number of turbo iterations was four. The performance of the TCM code with memory 6 utilising a channel interleaver was also plotted for comparison.

13.7.3.3.3 Fixed-mode Performance Before characterising the proposed wideband BbB adaptive scheme, the BER performance of the fixed modem modes of 4QAM, 8PSK, 16QAM and 64QAM are studied both with and without channel interleavers. These results are shown in Figure 13.41 for TCM, and in Figure 13.42 for TTCM. The random TTCM symbol-interleaver memory was set to 684 symbols, corresponding to the number of data symbols in the transmission burst structure of Figure 13.38, where the resultant number of bits was the number of data bits per symbol $(BPS) \times 684$. A channel interleaver of 4×684 symbols was utilised, where the number of bits was $(BPS + 1) \times 4 \times 684$ bits, since one parity bit was added to each TCM/TTCM symbol.

As expected, in Figures 13.41 and 13.42 the BER performance of the channel-interleaved scenario was superior compared to that without channel interleaver, although at the cost of an associated higher transmission delay. The SNR gain difference between the channel-interleaved and non-interleaved scenarios was about 5 dB in the TTCM/4QAM mode, but this difference reduced for higher-order modulation modes. Again, this gain was obtained at the cost of a four-burst channel interleaving delay. This SNR gain difference shows the importance of time diversity in coded modulation schemes.

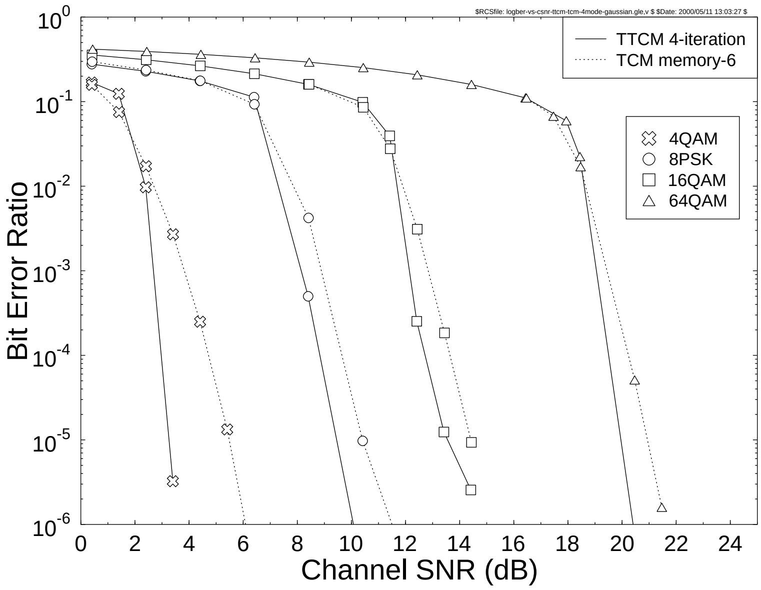


Figure 13.43: TTCM and TCM performance of each individual modulation mode for transmissions over the unfaded COST207 TU channel of Figure 13.37. The TTCM scheme used component TCM codes of memory 3 and the number of turbo iterations was four. The performance of the TCM scheme in conjunction with memory 6 was plotted for comparison with the similar-complexity TTCM scheme.

TTCM has been shown to be more efficient than TCM for transmissions over AWGN channels and narrowband fading channels [92, 326]. Here, we illustrate the advantage of TTCM in comparison to TCM over the dispersive or wideband Gaussian CIR of Figure 13.37, as seen in Figure 13.43. In conclusion, TTCM is superior to TCM in a variety of channels.

Let us now compare the performance of the BbB adaptive TCM/TTCM **system I** and **II**.

13.7.3.3.4 System I and System II Performance The modem mode switching mechanism of the adaptive schemes is characterised by a set of switching thresholds, the corresponding random TTCM symbol interleavers and the component codes, as follows:

$$\text{Modulation Mode} = \begin{cases} 4QAM, I_0 = 684, R_0 = 1/2 & \text{if } \gamma_{DFE} \leq f_1 \\ 8PSK, I_1 = 1368, R_1 = 2/3 & \text{if } f_1 < \gamma_{DFE} \leq f_2 \\ 16QAM, I_2 = 2052, R_2 = 3/4 & \text{if } f_2 < \gamma_{DFE} \leq f_3 \\ 64QAM, I_3 = 3420, R_3 = 5/6 & \text{if } \gamma_{DFE} > f_3, \end{cases} \tag{13.55}$$

where $f_n, n = 1 \ldots 3$, are the equaliser's output SNR thresholds, while I_n represents the random TTCM symbol interleaver size in terms of the number of bits, which is not used for the TCM schemes. The switching thresholds f_n were chosen experimentally, in order to maintain a BER of below 0.01%, and these thresholds are listed in Table 13.8.

BER $<$ 0.01 % Adaptive System Type		Switching Thresholds f_1	f_2	f_3
TCM, Memory 3	System I	19.56	23.91	30.52
	System II	17.17	21.91	29.61
TCM, Memory 6	System I	19.56	23.88	30.07
	System II	17.14	21.45	29.52
TTCM, 4 iterations	System I	19.69	23.45	30.29
	System II	16.66	21.40	28.47
BICM, Memory 3	System I	19.94	24.06	31.39
BICM-ID, 8 iterations	System II	16.74	21.45	28.97

Table 13.8: The switching thresholds were set experimentally for transmissions over the COST207 TU channel of Figure 13.37, in order to achieve a target BER of below 0.01%. **System I** does not utilise a channel interleaver, while **System II** uses a channel interleaver length of four TDMA/TDD bursts.

Let us consider the adaptive TTCM scheme in order to investigate the performance of **System I** and **System II**. The performance of the non-interleaved **System I** was found to be identical to that of the same scheme employing interleaving over one transmission burst, which was described in the context of Figure 13.38. This is because in the context of the burst-invariant fading scenario the channel behaves like a dispersive Gaussian channel, encountering a specific fading envelope and phase trajectory across a transmission burst. The CIR is then faded at the end or at the commencement of each transmission burst. Hence the employment of a channel interleaver having a memory of one transmission burst would not influence the distribution of the channel errors experienced by the decoder. The BER and BPS performances of both adaptive TTCM systems using four iterations are shown in Figure 13.44, where we observed that the throughput of **System II** was superior to that of **System I**. Furthermore, the overall BER of **System II** was lower than that of **System I**. In order to investigate the switching dynamics of both systems, the mode switching together with the equaliser's output SNR was plotted versus time at an average channel SNR of 25 dB in Figures 13.45 and 13.46. Observe in Table 13.8 that the switching thresholds f_n of **System II** are lower than those of **System I**, since the fixed-mode-based results of **System II** in Figure 13.42 were better. Hence higher-order modulation modes were chosen more frequently than in **System I**, giving a better BPS throughput. From Figures 13.45 and 13.46, it is clear that **System I** was more flexible in terms of mode switching, while **System II** benefited from higher diversity gains due to the four-burst channel interleaver. This diversity gain compensated for the loss of switching flexibility, ultimately providing a better performance in terms of BER and BPS, as seen in Figure 13.44.

In our next endeavour, the adaptive TCM and TTCM schemes of **System I** and **System II** are compared. Figure 13.47 shows the BER and BPS performance of **System I** for adaptive TTCM using four iterations, adaptive TCM of memory 3 (which was the component code of our TTCM scheme) and adaptive TCM of memory 6 (which had a similar decoding complexity to our TTCM scheme). As can be seen from the fixed-mode results of Figures 13.41 and 13.42 in the previous section, TCM and TTCM performed similarly in terms of their BER, when no channel interleaver was used for the slow fading wideband COST207 TU channel of Figure 13.37. Hence, they exhibited a similar performance in the context of the adaptive

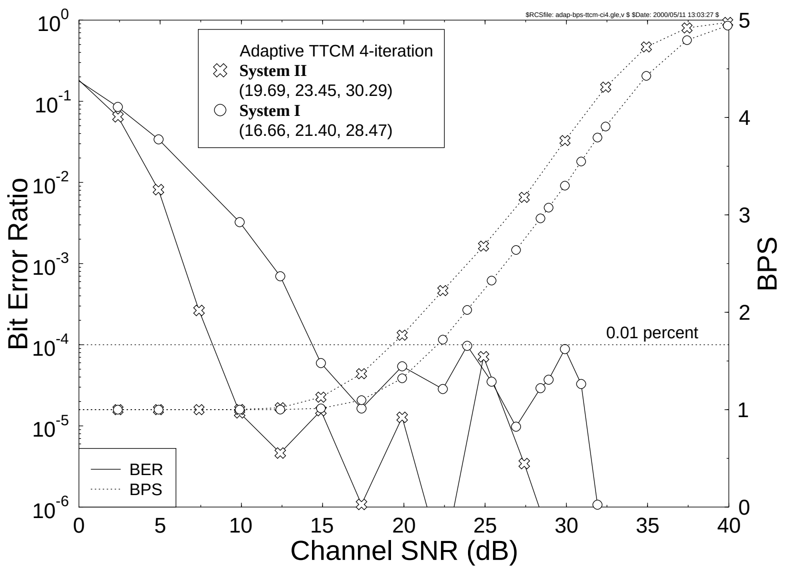


Figure 13.44: BER and BPS performance of adaptive TTCM for transmissions over the COST207 TU channel of Figure 13.37, using four turbo iterations in **System I** (without channel interleaver) and in **System II** (with a channel interleaver length of four bursts) for a target BER of less than 0.01%. The legends indicate the associated switching thresholds expressed in dB, as seen in the brackets.

schemes of System I, as shown in Figure 13.47. Even the TCM scheme of memory 3 associated with a lower complexity could give a similar BER and BPS performance. This shows that the equaliser plays a dominant role in **System I**, where the coded modulation schemes could not benefit from sufficient diversity due to the lack of interleaving.

When the channel interleaver is introduced in **System II**, the bursty symbol errors are dispersed. Figure 13.48 illustrates the BER and BPS performance of **System II** for adaptive TTCM using four iterations, adaptive TCM of memory 3 and adaptive TCM of memory 6. The performance of all these schemes improved in the context of **System II**, as compared to the corresponding schemes in **System I**. The TCM scheme of memory 6 had a lower BER than TCM of memory 3, and also exhibited a small BPS improvement. As expected, TTCM had the lowest BER and also the highest BPS throughput compared to the other coded modulation schemes.

In summary, we have observed BER and BPS gains for the channel-interleaved adaptive coded schemes of **System II** in comparison to the schemes without channel interleaver as in **System I**. Adaptive TTCM exhibited a superior performance in comparison to adaptive TCM in the context of **System II**.

13.7.3.3.5 Overall Performance Figure 13.49 shows the fixed modem modes' performance for TCM, TTCM, BICM and BICM-ID in the context of **System II**. For the sake of a fair comparison of the DC, we used a TCM code memory of 6, TTCM code memory of 3 in conjunction with four turbo iterations, BICM code memory of 6 and a BICM-ID code memory of 3 in conjunction with eight decoding iterations. However, BICM-ID had a slightly higher DC, since the demodulator was invoked in each BICM-ID iteration, whereas in the BICM, TCM and TTCM schemes the demodulator was only visited once in each decoding process. As illustrated in the figure, the BICM scheme performed marginally better than the TCM scheme at a BER below 0.01%, except in the 64QAM mode. Hence, adaptive BICM is also expected to be better than adaptive TCM in the context of **System II**, when a target BER of less than 0.01% is desired. This is because when the channel interleaver depth is sufficiently high, the diversity gain of the BICM's bit interleaver is

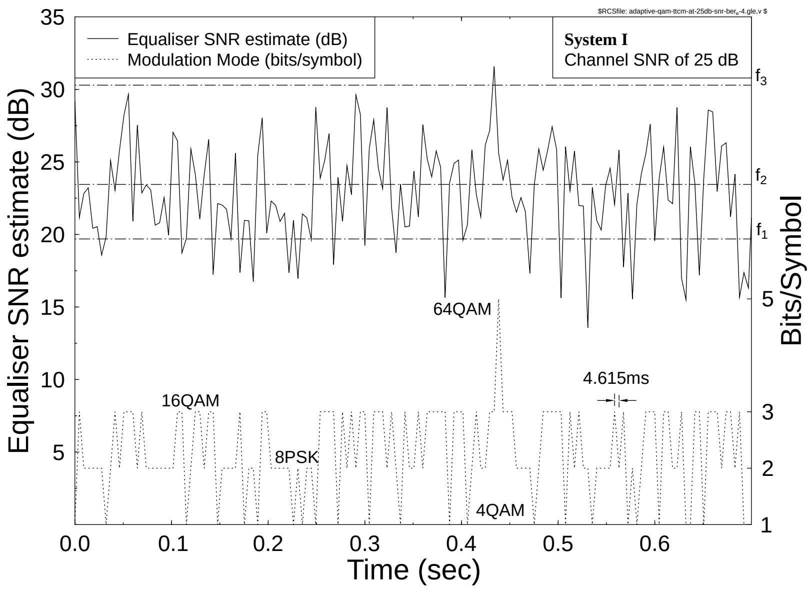


Figure 13.45: Channel SNR estimate and BPS versus time plot for adaptive TTCM for transmissions over the COST207 TU channel of Figure 13.37, using four turbo iterations in **System I** at an average channel SNR of 25 dB, where the modulation mode switching is based upon the equaliser's output SNR, which is compared to the switching thresholds f_n defined in Table 13.8. The duration of one TDMA/TDD frame is 4.615 ms. The TTCM mode can be switched after one frame duration.

higher than that of the TCM's symbol interleaver [84, 91].

Figure 13.50 compares the adaptive BICM and TCM schemes in the context of **System I**, i.e. without channel interleaving, although the BICM scheme invoked an internal bit interleaver of one burst memory. As can be seen from the figure, adaptive TCM exhibited a better BPS throughput and BER performance than BICM, due to employing an insufficiently high channel interleaving depth for the BICM scheme, for transmissions over our slow fading wideband channels.

As observed in Figure 13.49, we noticed that BICM-ID had the worst performance at low SNRs in each modulation mode compared to other coded modulation schemes. However, it exhibited a steep slope and therefore at high SNRs it approached the performance of the TTCM scheme. The adaptive BICM-ID and TTCM schemes employed in the context of **System II** were compared in Figure 13.50. The adaptive TTCM scheme exhibited a better BPS throughput than adaptive BICM-ID, since TTCM had a better performance in fixed modem modes at a BER of 0.01%. However, adaptive BICM-ID exhibited a lower BER performance than adaptive TTCM owing to the high steepness of the BER curve of BICM-ID in its fixed modem modes.

13.7.3.3.6 Conclusions In this section, BbB adaptive TCM, TTCM, BICM and BICM-ID were proposed for transmissions over wideband fading channels both with and without channel interleaving and they were characterised in performance terms when communicating over the COST207 TU fading channel. When observing the associated BPS curves, adaptive TTCM exhibited up to 2.5 dB SNR gain for a channel interleaver length of four bursts in comparison to the non-interleaved scenario, as evidenced in Figure 13.44. Upon comparing the BPS curves, adaptive TTCM also exhibited up to 0.7 dB SNR gain compared to adaptive TCM of the same complexity in the context of **System II** for a target BER of less than 0.01%, as shown in Figure 13.48. Lastly, adaptive TCM performed better than the adaptive BICM bench-marker in the context of **System I**, and the adaptive BICM-ID scheme performed marginally worse than adaptive

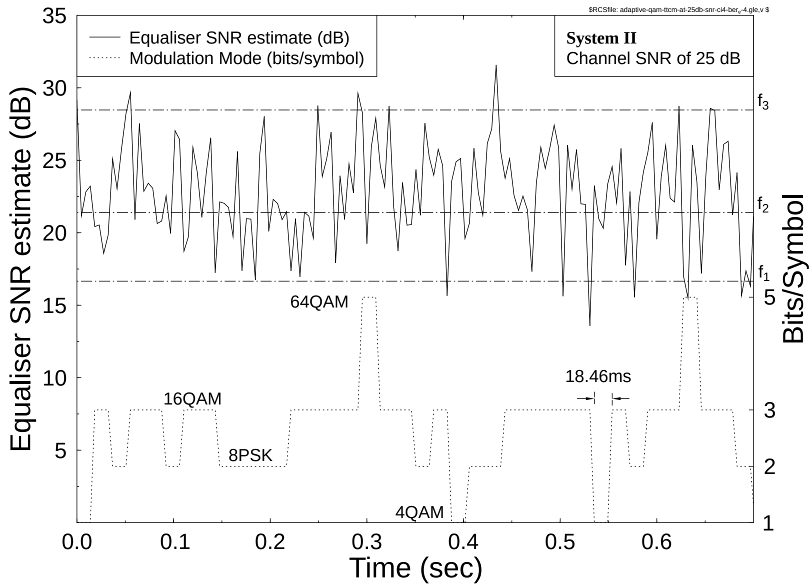


Figure 13.46: Channel SNR estimate and BPS versus time plot for adaptive TTCM for transmissions over the COST207 TU channel of Figure 13.37, using four turbo iterations in **System II** at an average channel SNR of 25 dB, where the modulation mode switching is based upon the equaliser's output SNR which is compared to the switching thresholds f_n defined in Table 13.8. The duration of one TDMA/TDD frame is 4.615 ms. The TTCM mode is maintained for four frame durations, i.e. for 18.46 ms.

TTCM in the context of **System II** as discussed in Section 13.7.3.3.5.

13.7.3.4 Orthogonal Frequency Division Multiplexing

The employment of equalisers for removing the ISI has been discussed in Section 13.7.3.2. By contrast, as an attractive design alternative here Discrete Fourier Transform (DFT)-based Orthogonal Frequency Division Multiplexing (OFDM) [193] will be utilised for removing the ISI.

13.7.3.4.1 Orthogonal Frequency Division Multiplexing Principle In this section we briefly introduce Frequency Division Multiplexing (FDM), also referred to as Orthogonal Multiplexing (OMPX), as a means of dealing with the problems of frequency-selective fading encountered when transmitting over a high-rate wideband radio channel. The fundamental principle of orthogonal multiplexing originates from Chang [327], and over the years a number of researchers have investigated this technique [328, 329]. Despite its conceptual elegance, until recently its employment has been mostly limited to military applications because of the associated implementational difficulties. However, it has recently been adopted as the new European Digital Audio Broadcasting (DAB) standard; it is also a strong candidate for Digital Terrestrial Television Broadcast (DTTB) and for a range of other high-rate applications, such as 155 Mbit/s wireless Asynchronous Transfer Mode (ATM) local area networks. These wide-ranging applications underline its significance as an alternative technique to conventional channel equalisation in order to combat signal dispersion [219, 330, 331].

In the FDM scheme of Figure 13.51 the serial data stream of a traffic channel is passed through a serial-to-parallel converter, which splits the data into a number of parallel sub-channels. The data in each sub-channel are applied to a modulator, such that for N channels there are N modulators whose carrier

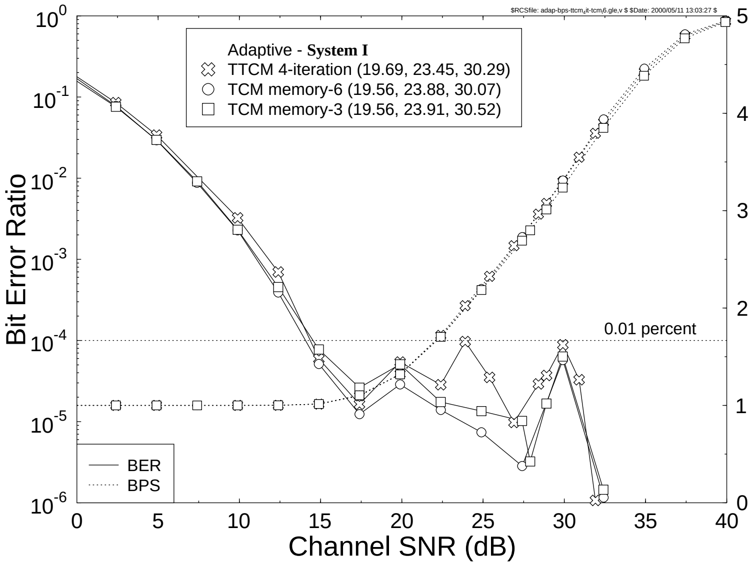


Figure 13.47: BER and BPS performance of adaptive TCM and TTCM without channel interleaving in **System I**, for transmissions over the Rayleigh fading COST207 TU channel of Figure 13.37. The switching mechanism was characterised by Equation 13.55. The switching thresholds were set experimentally, in order to achieve a BER of below 0.01%, as shown in Table 13.8.

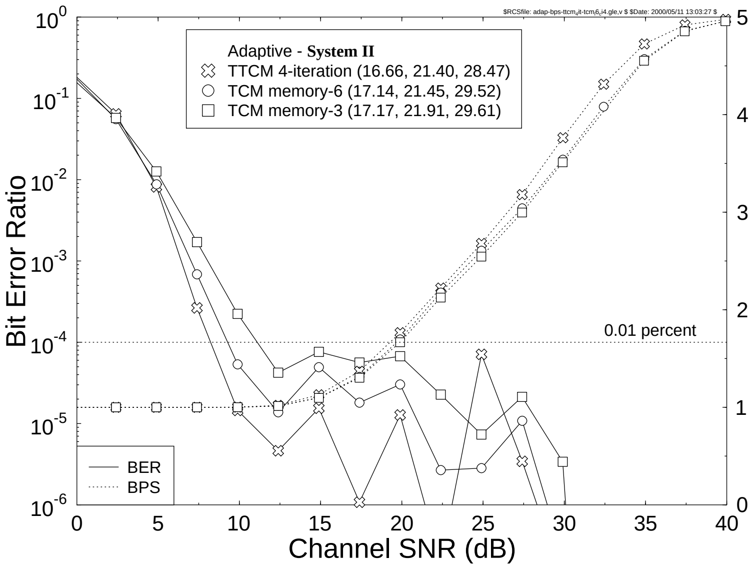


Figure 13.48: BER and BPS performance of adaptive TCM and TTCM using a channel interleaver length of four bursts in **System II**, for transmissions over the Rayleigh fading COST207 TU channel of Figure 13.37. The switching mechanism was characterised by Equation 13.55. The switching thresholds were set experimentally, in order to achieve a BER of below 0.01%, as shown in Table 13.8.

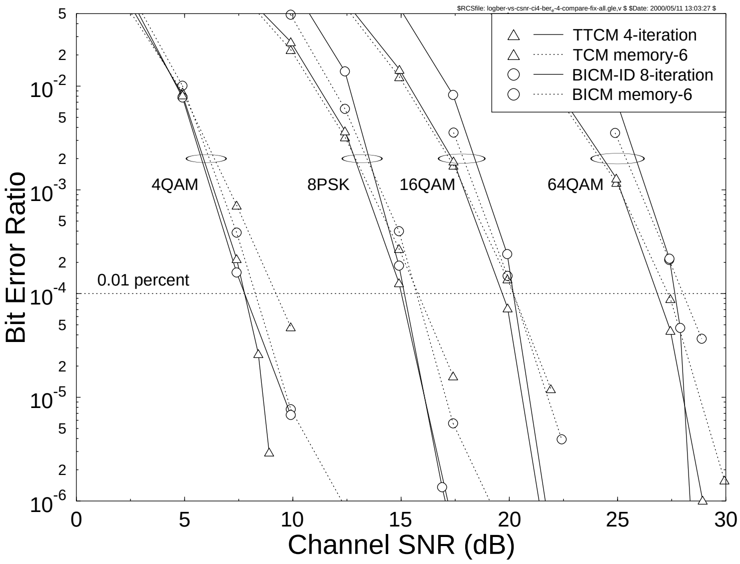


Figure 13.49: BER performance of the fixed modem modes of 4QAM, 8PSK, 16QAM and 64QAM utilising TCM, TTCM, BICM and BICM-ID schemes in the context of **System II** for transmissions over the COST207 TU channel of Figure 13.37. For the sake of maintaining a similar DC, we used a TCM code memory of 6, TTCM code memory of 3 in conjunction with four turbo iterations, BICM code memory of 6 and a BICM-ID code memory of 3 in conjunction with eight decoding iterations. However, BICM-ID had a slightly higher complexity than the other systems, since the demodulator module was invoked eight times as compared to only once for its counterparts during each decoding process.

frequencies are f_0, f_1, ..., f_{N-1}. The centre frequency difference between adjacent channels is Δf and the overall bandwidth W of the N modulated carriers is $N\Delta f$.

These N modulated carriers are then combined to give a FDM signal. We may view the serial-to-parallel converter as applying every Mth symbol to a modulator. This has the effect of interleaving the symbols entered into each modulator, hence symbols $S_0, S_N, S_{2N}, \ldots$ are applied to the modulator whose carrier frequency is f_1. At the receiver the received FDM signal is demultiplexed into N frequency bands, and the N modulated signals are demodulated. The baseband signals are then recombined using a parallel-to-serial converter.

The main advantage of the above FDM concept is that because the symbol period has been increased, the channel's delay spread is a significantly shorter fraction of a symbol period than in the serial system, potentially rendering the system less sensitive to ISI than the conventional N times higher-rate serial system. In other words, in the low-rate sub-channels the signal is no longer subject to frequency-selective fading, hence no channel equalisation is necessary.

A disadvantage of the FDM approach shown in Figure 13.51 is its increased complexity in comparison to the conventional system caused by employing N modulators and filters at the transmitter and N demodulators and filters at the receiver. It can be shown that this complexity can be reduced by employing the Discrete Fourier Transform (DFT), typically implemented with the aid of the Fast Fourier Transform (FFT) [193]. The sub-channel modems can use almost any modulation scheme, and 4- or 16-level QAM is an attractive choice in many situations.

The FFT-based QAM/FDM modem's schematic is portrayed in Figure 13.52. The bits provided by the source are serial/parallel converted in order to form the n-level Gray-coded symbols, N of which are collected in TX buffer 1, while the contents of TX buffer 2 are being transformed by the Inverse Fast

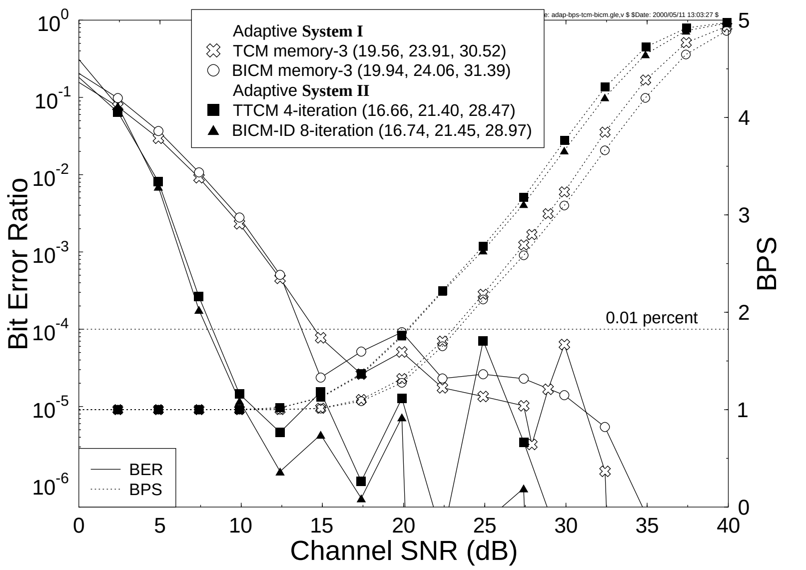


Figure 13.50: BER and BPS performance of the adaptive TCM/BICM **System I**, using memory 3 codes and that of the adaptive TTCM/BICM-ID **System II**, for transmissions over the Rayleigh fading COST207 TU channel of Figure 13.37. The switching mechanism was characterised by Equation 13.55. The switching thresholds were set experimentally, in order to achieve a BER of below 0.01%, as shown in Table 13.8.

Fourier Transform (IFFT) in order to form the time-domain modulated signal. The Digital-to-Analogue (D/A) converted, low-pass filtered modulated signal is then transmitted via the channel and its received samples are collected in RX buffer 1, while the contents of RX buffer 2 are being transformed to derive the demodulated signal. The twin buffers are alternately filled with data to allow for the finite FFT-based demodulation time. Before the data are Gray coded and passed to the data sink, they can be equalised by a low-complexity method, if there are some dispersions within the narrow sub-bands. For a deeper tutorial exposure the interested reader is referred to reference [193].

13.7.3.5 Orthogonal Frequency Division Multiplexing Aided Coded Modulation

13.7.3.5.1 Introduction The coded modulation schemes of Sections 13.2–13.6 are here integrated with OFDM by mapping coded symbols to the OFDM modulator at the transmitter and channel decoding the symbols output by the OFDM demodulator at the receiver.

When the channel is frequency selective and OFDM modulation is used, the received symbol is given by the product of the transmitted frequency-domain OFDM symbol and the frequency-domain transfer function of the channel. This direct relationship facilitates the employment of simple frequency-domain channel equalisation techniques. Essentially, if an estimate of the complex frequency-domain transfer function H_k is available at the receiver, channel equalisation can be performed by dividing each received value by the corresponding frequency-domain channel transfer function estimate. The channel's frequency-domain transfer function can be estimated with the aid of known frequency-domain pilot subcarriers inserted into the transmitted signal's spectrum [193]. These known pilots effectively sample the channel's frequency-domain transfer function according to the Nyquist frequency. These frequency-domain samples then allow us to recover the channel's transfer function between the frequency-domain pilots with the aid of interpolation. In addition to this simple form of channel equalisation, another advantage of the OFDM-based modulation is that it turns a channel exhibiting memory into a memoryless one, where the memory is the influence of

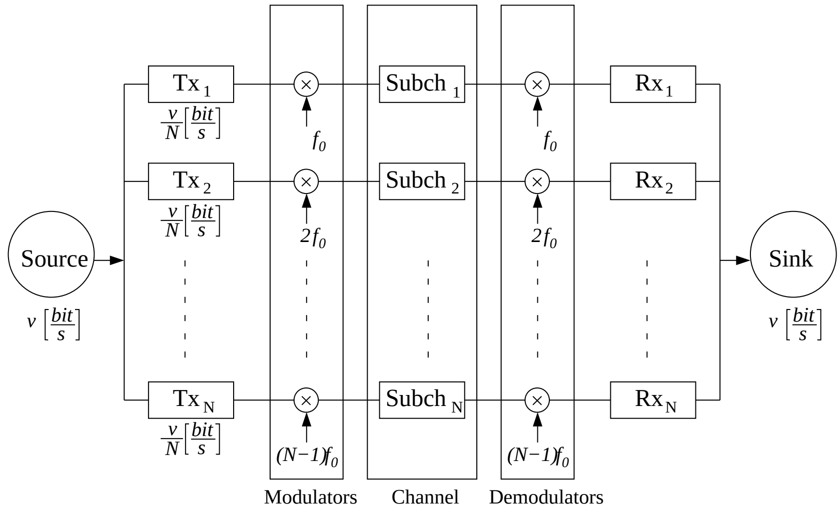


Figure 13.51: Simplified block diagram of the orthogonal parallel modem.

the past transmitted symbols on the value of the present symbol.

13.7.3.5.2 System Overview The encoder produces a block of N_i channel symbols to be transmitted. These symbols are transmitted by the OFDM modulator. As the OFDM modulator transmits N_u modulated symbols per OFDM symbol, if $N_u = N_i$ then the whole block of N_u modulated symbols can be transmitted in a single OFDM symbol. We refer to this case as the single-symbol mapping based scenario. By contrast, if $N_u < N_i$, then more than one OFDM symbol is required for the transmission of the channel-coded block. We refer to this case as the multiple-symbol mapping based scenario. Both the single- and the multiple-symbol scenarios are interesting. The single-symbol scenario is more appealing from an implementation point of view, as it is significantly more simple. However, it is well known that the performance of a turbo-coded scheme improves upon increasing the IL. Since the number of subcarriers in an OFDM system is limited by several factors, such as for example the oscillator's frequency stability, by using the single-symbol solution we would be limited in terms of the TTCM block length as well. Thus the multiple-symbol solution also has to be considered in order to fully exploit the advantages of the TTCM scheme. Since the single-symbol based scheme is conceptually more simple, we will consider this scenario first. Its extension to the multiple-symbol scenario is straightforward.

In the single carrier system discussed in Section 13.7.3.2 the received signal is given by $y_k = x_k * h_k + n_k$, where $*$ denotes the convolution of the transmitted sequence x_k with the channel's impulse response h_k. An equaliser is used for removing the ISI before channel decoding, giving $\tilde{y}_k = \tilde{x}_k + \tilde{n}_k$. The associated branch metrics can be computed as:

$$\begin{aligned} P(\tilde{y}_k|\tilde{x}_k) &= e^{-\frac{|\tilde{n}|^2}{2\sigma^2}} \\ &= e^{-\frac{|\tilde{y}_k - \tilde{x}_k|^2}{\sigma^2}}. \end{aligned} \tag{13.56}$$

However, in a multi-carrier OFDM system, the received signal is given by $Y_k = X_k \cdot H_k + N_k$, which facilitates joint channel equalisation and channel decoding by computing the branch metrics as:

$$P(y_k|x_k) = e^{-\frac{|Y_k - H_k \cdot X_k|^2}{2\sigma^2}}, \tag{13.57}$$

where H_k is the channel's frequency-domain transfer function at the centre frequency of the kth subcarrier. Hence, as long as the the channel transfer function estimation is of sufficiently high quality, simple

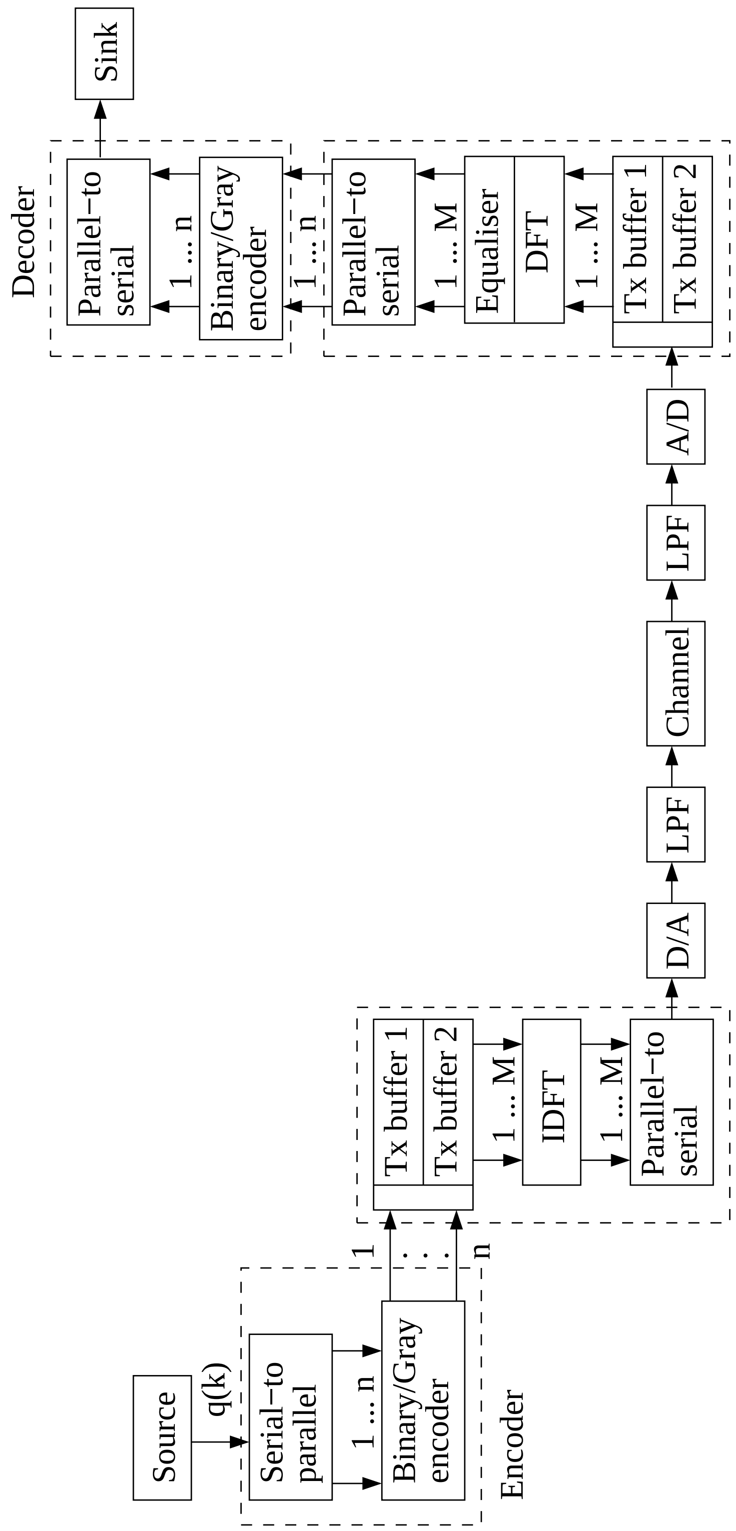


Figure 13.52: FFT-based OFDM modem schematic ©Webb, Hanzo, 1994, [193]

OFDM Parameters	
Total number of subcarriers, N	2048 (2K mode)
Number of effective subcarriers, N_u	1705
OFDM symbol duration T_s	224 μs
Guard interval	$T_s/4 = 56\mu$s
Total symbol duration (inc. guard interval)	280 μs
Consecutive subcarrier spacing $1/T_s$	4464 Hz
DVB channel spacing	7.61 MHz
QPSK and QAM symbol period	7/64 μs
Baud rate	9.14 MBaud

Table 13.9: Parameters of the OFDM module [333].

frequency-domain equalisation could be invoked during the decoding process. If iterative channel decoding is invoked, the channel transfer function estimation is expected to improve during the consecutive iterative steps, in a fashion known in the context of turbo equalisation [129]. Indeed, a performance as high as that in conjunction with perfect channel estimation can be attained [332].

Let us now consider the effect of the channel interleaver. When the channel is frequency selective, it exhibits frequency-domain nulls, which may obliterate several OFDM subcarriers. Thus the quality of several consecutive received OFDM symbols will be low. If the quality is inferior, the channel decoder is unable to correctly estimate the transmitted symbols. When, however, a channel interleaver is present, the received symbols are shuffled before channel decoding and hence these clusters of corrupted subcarriers are disperse. Thus, after the channel interleaver we expect to have only isolated low-quality subcarriers surrounded by unimpaired ones. In this case the decoder is more likely to be able to recover the symbol transmitted on the corrupted subcarrier, using the redundancy added by the channel coding process, conveyed by the surrounding unimpaired subcarriers.

Finally, we consider the multiple-symbol scenario. The system requires only a minor modification. When more than one OFDM symbol is required for transmitting the block of channel-coded symbols, the OFDM demodulator has to store both the demodulated symbols and the channel transfer function estimates in order to form a whole channel-coded block. This block is then fed to the channel interleaver and then to the channel decoder, together with the channel transfer function estimate. Exactly the same MAP decoder as in the single-symbol case can be used for performing the joint channel equalisation and channel decoding. Similarly, the function of the channel interleaver is the same as in the single-symbol scenario.

13.7.3.5.3 Simulation Parameters The Digital Video Broadcasting (DVB) standard's OFDM scheme [193] was used for this study. The parameters of the OFDM DVB system are presented in Table 13.9. Since the OFDM modem has 2048 subcarriers, the subcarrier signalling rate is effectively 2000 times lower than the maximum DVB transmission rate of 20 Mbit/s, corresponding to about 10 kbit/s. At this sub-channel rate, the individual sub-channels can be considered nearly frequency-flat.

The channel model employed is the 12-path COST207 [237] Hilly Terrain (HT) type of impulse response, exhibiting a maximum relative path delay of 19.9 μs. The unfaded impulse response is depicted in Figure 13.53. The carrier frequency is 500 MHz and the sampling rate is 7/64 μs. Each of the channel paths was faded independently, obeying a Rayleigh fading distribution, according to a normalised Doppler frequency of 10^{-5} [49]. This corresponds to a worst-case vehicular velocity of about 200 km/h.

13.7.3.5.4 Simulation Results and Discussions In this section, the system performance of the OFDM-based coded modulation schemes is evaluated using QPSK, 8PSK and 16QAM. The coding rate of the coded modulation schemes changes according to the modem mode used. Hence the effective throughput for QPSK is 1 bit/subcarrier, for 8PSK it is 2 bits/subcarrier and finally for 16QAM it is 3 bits/subcarrier.

Figure 13.54 shows the performance of the integrated systems in conjunction with a channel interleaver of 5000 symbols using a QPSK modem. We can see from the figure that BICM performs about 3.8 dB

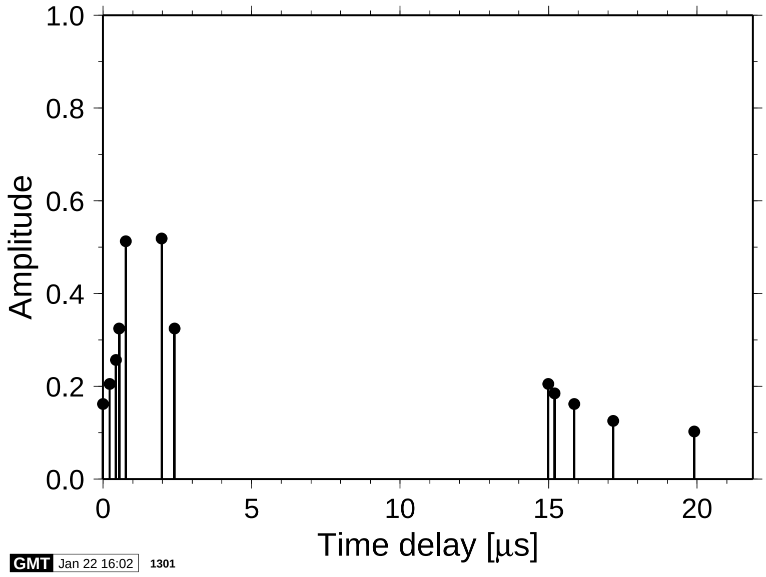


Figure 13.53: COST207 Hilly Terrain (HT) type of impulse response [237].

better than TCM at a BER of 10^{-4}. When the iterative schemes of TTCM and BICM-ID are used, a better performance is achieved. When using eight iterations, TTCM and BICM-ID reach their optimum performance. Clearly, in this scenario TTCM is superior to BICM-ID.

Figure 13.55 illustrates the performance of the systems considered for an effective throughput of 2 bits/subcarrier using a channel interleaver of 5000 symbols. TCM, BICM and BICM-ID exhibit a similar complexity; however, TTCM using eight iterations is somewhat more complex. Specifically, in order to be able to compare the associated complexities we assumed that the number of decoder trellis states determined the associated complexity. For example, a memory-length $K = 3$ TTCM scheme had $2^K = 8$ trellises per decoding iteration and there are two decoders. Hence after four iterations we encounter a total of $8 \cdot 2 \cdot 4 = 64$ trellis states. Hence a 64-state, $K = 6$ TCM scheme has a similar complexity to a $K = 4$ TTCM arrangement using four iterations. For the sake of fair comparisons, TTCM employing four iterations should be used, although the performance difference between four and eight iterations is only marginal. Again, TTCM is the best scheme. At a BER of 10^{-4}, TTCM performs about 1.2 dB better than BICM-ID, 2.2 dB better than BICM and 3.6 dB better than TCM.

Figure 13.56 compares the performance of the systems for a throughput of 3 bits/subcarrier using a channel interleaver of 5000 symbols. Again, the systems exhibited a similar complexity, except for TTCM. There is one uncoded information bit in the 4-bit 16QAM symbol of the rate-3/4 TCM code having 64 states, as can be seen from its generator polynomial in Table 13.6. For this reason, this TCM scheme is only potent at lower SNRs, while exhibiting modest performance improvements in the higher SNR region, as demonstrated by Figure 13.56. The rest of the schemes do not have uncoded information bits in their 16QAM symbols. BICM-ID outperforms BICM for E_b/N_0 values in excess of about 1.2 dB, while it is inferior in comparison to TTCM by about 1 dB at a BER of 10^{-4}. However, TTCM using 8-state TCM component codes exhibits an error floor at a BER around 10^{-5}.

13.7.3.5.5 Conclusions In this section OFDM was studied and integrated with the coded modulation schemes of Sections 13.2–13.6. The performance of OFDM-assisted TCM, TTCM, BICM and BICM-ID was investigated for transmissions over the dispersive COST207 HT Rayleigh fading channel of Figure 13.53 using QPSK, 8PSK and 16QAM modulation modes. TTCM was found to be the best compromise scheme, followed by BICM-ID, BICM and TCM.

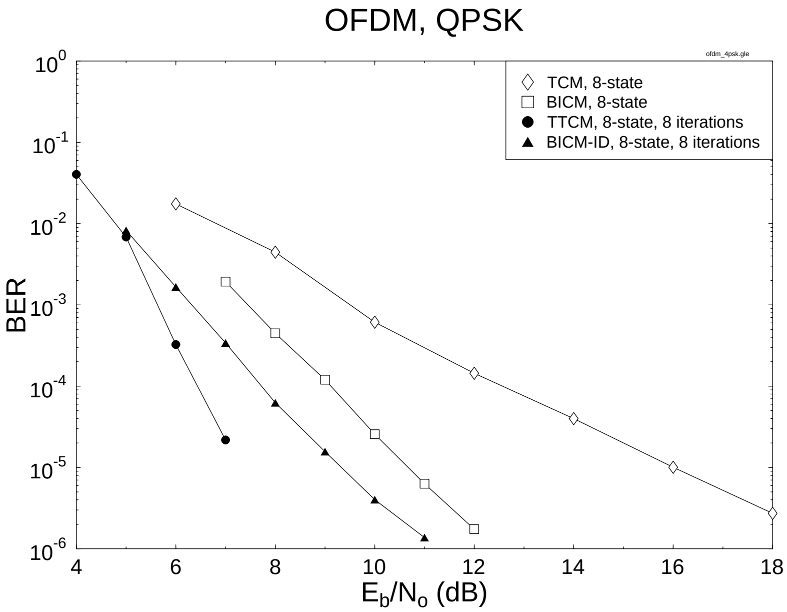


Figure 13.54: Comparison of TCM, TTCM, BICM and BICM-ID using QPSK-OFDM modem for transmissions over the Rayleigh fading COST207 HT channel of Figure 13.53 at a normalised Doppler frequency of 10^{-5}. The OFDM parameters are listed in Table 13.9, while the coded modulation parameters were summarised in Section 13.7.2.1.

13.8 Summary and Conclusions

In Sections 13.2–13.6 we have studied the conceptual differences between four coded modulation schemes in terms of their coding structure, signal labelling philosophy, interleaver type and decoding philosophy. The symbol-based non-binary MAP algorithm was also highlighted, when operating in the logarithmic domain.

Furthermore, in Section 13.7 the performance of the above-mentioned four coded modulation schemes was evaluated for transmissions over narrowband channels in Section 13.7.2 and over wideband channels in Section 13.7.3. Over the dispersive Rayleigh fading channels a DFE was utilised for supporting the operation of the coded modulation schemes, as investigated in Section 13.7.3.3.

TCM was found to perform better than BICM in AWGN channels owing to its higher Euclidean distance, while BICM faired better in fading channels, since it was designed for fading channels with a higher grade of time diversity in mind. BICM-ID gave better performance both in AWGN and fading channels compared to TCM and BICM, although it exhibited a higher complexity due to employing several decoding iterations, while TTCM struck the best balance between performance and complexity.

The performance of the BbB DFE-assisted adaptive coded modulation scheme was investigated in Section 13.7.3.3, and attained an improved performance in comparison to the fixed-mode-based coded modulation schemes in terms of both its BER and BPS performance.

OFDM was also invoked for assisting the operation of coded modulation schemes in highly dispersive propagation environments, as investigated in Section 13.7.3.5 in a multi-carrier transmission scenario. Again, TTCM/OFDM struck the most attractive trade-off in terms of its performance versus complexity balance in these investigations, closely followed by BICM-ID/OFDM.

Having introduced a host of channel coding techniques in the previous chapters, in the next two chapters we will concentrate on the family of transmit-diversity-aided space-time coding arrangements, which are very powerful in terms of mitigating the effects of fading when communicating over wireless channels.

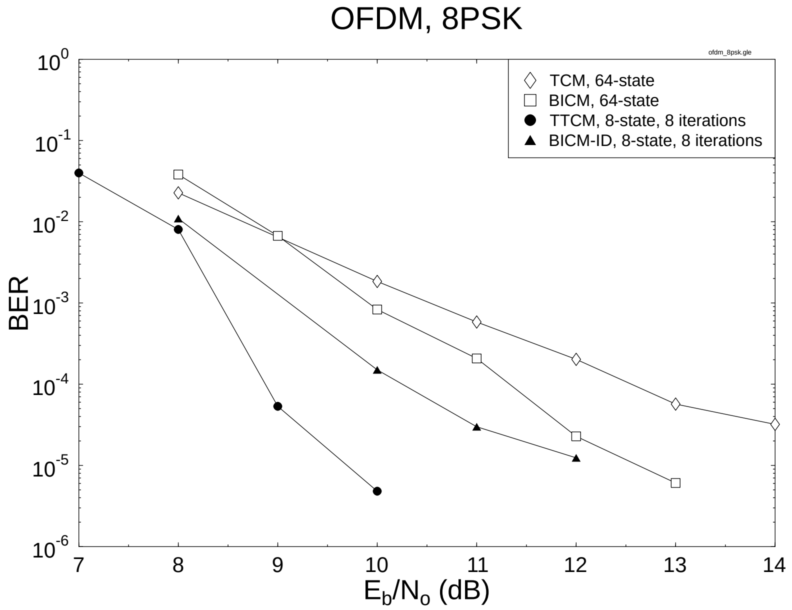


Figure 13.55: Comparison of TCM, TTCM, BICM and BICM-ID using 8PSK-OFDM modem for transmissions over the Rayleigh fading COST207 HT channel of Figure 13.53 at a normalised Doppler frequency of 10^{-5}. The OFDM parameters are listed in Table 13.9 while the coded modulation parameters were summarised in Section 13.7.2.1.

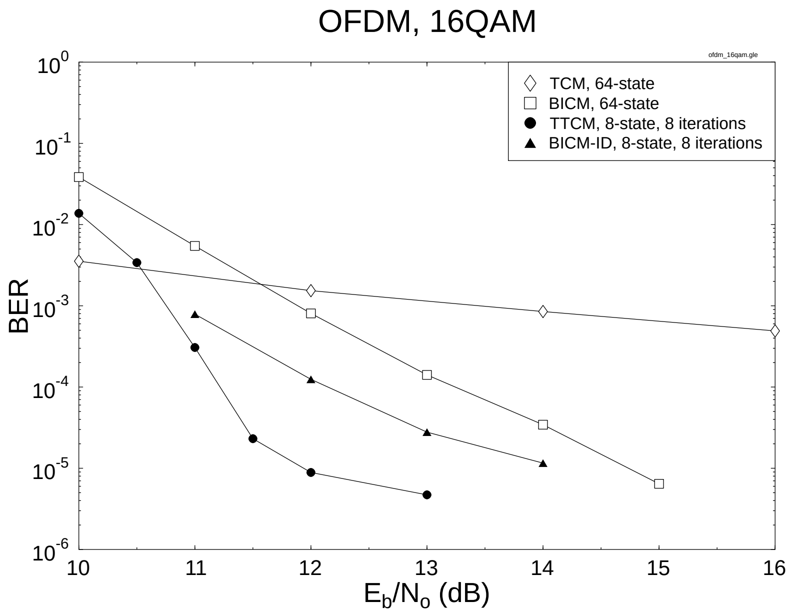


Figure 13.56: Comparison of TCM, TTCM, BICM and BICM-ID using 16QAM-OFDM modem for transmissions over the Rayleigh fading COST207 HT channel of Figure 13.53 at a normalised Doppler frequency of 10^{-5}. The OFDM parameters are listed in Table 13.9 while the coded modulation parameters were summarised in Section 13.7.2.1.

Chapter 18

Near-Capacity Irregular BICM-ID Scheme[1]

18.1 Introduction

In the previous chapters we have investigated various MLC and BICM-ID designs for the sake of improving their BER performance. In this chapter we further improve the BICM-ID scheme for the sake of achieving an infinitesimally low BER close to the theoretical capacity limit. In view of this, we introduce the irregular code-design concept contrived for BICM-ID schemes in order to create a more flexible system, which is capable of achieving a near-capacity performance. Alternatively, with the aid of these irregular components, we strike a more attractive trade off between near-capacity performance and a moderate system complexity.

BICM-ID [90] was described in Section 14.4, which consists of three main blocks - a convolutional encoder, bit-interleavers and a bit-to-symbol mapper. Iterative detection was achieved by exchanging soft extrinsic information between the symbol-to-bit demapper and the decoder. The BER versus E_b/N_0 curve of this BICM-ID scheme exhibits a turbo-like cliff and the achievable convergence performance can be characterised with the aid of EXIT charts [359].

Different bit-to-symbol mapping schemes have been investigated in order to improve the convergence behaviour of BICM-ID [415, 416] by shaping the demapper's EXIT characteristic for the sake of creating an open EXIT tunnel and hence to achieve an infinitesimally low BER. Furthermore, an adaptive BICM arrangement using various iterative decoding schemes was proposed in [417], which employed different signal constellations and bit-to-symbol mapping within one codeword. This flexible signalling scheme required the knowledge of the average signal quality at the transmitter, for invoking advanced Adaptive Modulation and Coding (AMC) [418] in order to improve the overall BICM-ID scheme's achievable performance.

A Unity-Rate Code (URC) can be used as a precoder for reaching the (1,1) EXIT chart convergence point [373] and hence to achieve an infinitesimally low BER. A precoded mapper scheme was also proposed for a three-stage - encoder, precoder and modulator - design constituted by either a bit-based or a symbol-based precoder [419]. Furthermore, a flexible irregular demapper combined with modulation doping was proposed in [420] for the sake of producing an open EXIT tunnel.

The Irregular Convolutional Coding (IrCC) [421, 422] concept was proposed for improving the convergence behaviour of serially concatenated codes, which can be employed as the encoder component of an Ir-BICM-ID. Furthermore, in order to introduce a more diverse set of EXIT characteristics for creating an IrCC, we invoke low-complexity Convolutional Codes (CC) constituted from a hybrid combination of memoryless repetition codes.

[1] Part of this chapter is based on the collaborative research outlined in [142, 143].

The novel contribution of this chapter is that we propose an irregular BICM-ID arrangement for the sake of creating an adaptive BICM-ID scheme, which can either exhibit near-capacity performance or lower-complexity implementation. Our approach is based on invoking EXIT chart analysis for minimising the area of the open EXIT tunnel in order to achieve a near-capacity performance. Alternatively, the width of the EXIT tunnel can be increased for the sake of reducing the system's complexity.

18.2 Irregular Bit-Interleaved Coded Modulation Schemes

The classic BICM-ID scheme of Section 14.4 exchanges extrinsic information between the CC decoder and the demapper, as shown again in Figure 18.1 for convenience. The precoder concept was proposed in [361] which was then used to create a recursive demapper at the decoder, as previously described in Section 15.4. By combining the precoder and the demapper, we will demonstrate that the demapper will facilitate convergence to the (1,1) EXIT trajectory point. The system's schematic is shown in Figure 18.2.

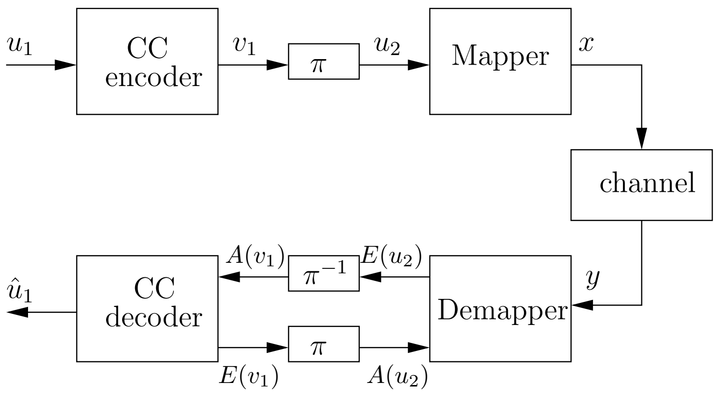


Figure 18.1: The general structure of the classic BICM-ID scheme. The binary source bit stream u_1 is encoded by the CC encoder and the encoded bits v_1 are interleaved by the bit interleaver π, yielding the permuted bits u_2. The inner iterations exchange extrinsic information between the demapper and the CC decoder, where the notation $A(.)$ represents the $a\ priori$ information quantified in terms of LLRs, while $E(.)$ denotes the $extrinsic$ information also expressed in terms of LLRs.

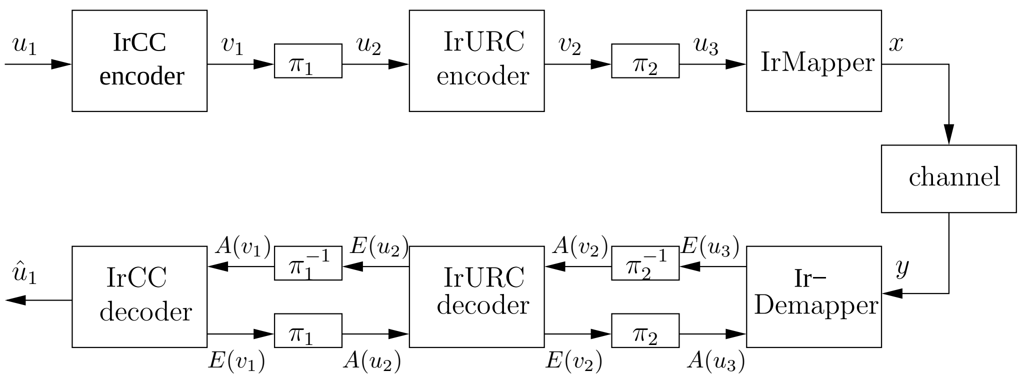


Figure 18.2: The general structure of the Ir-BICM-ID scheme.

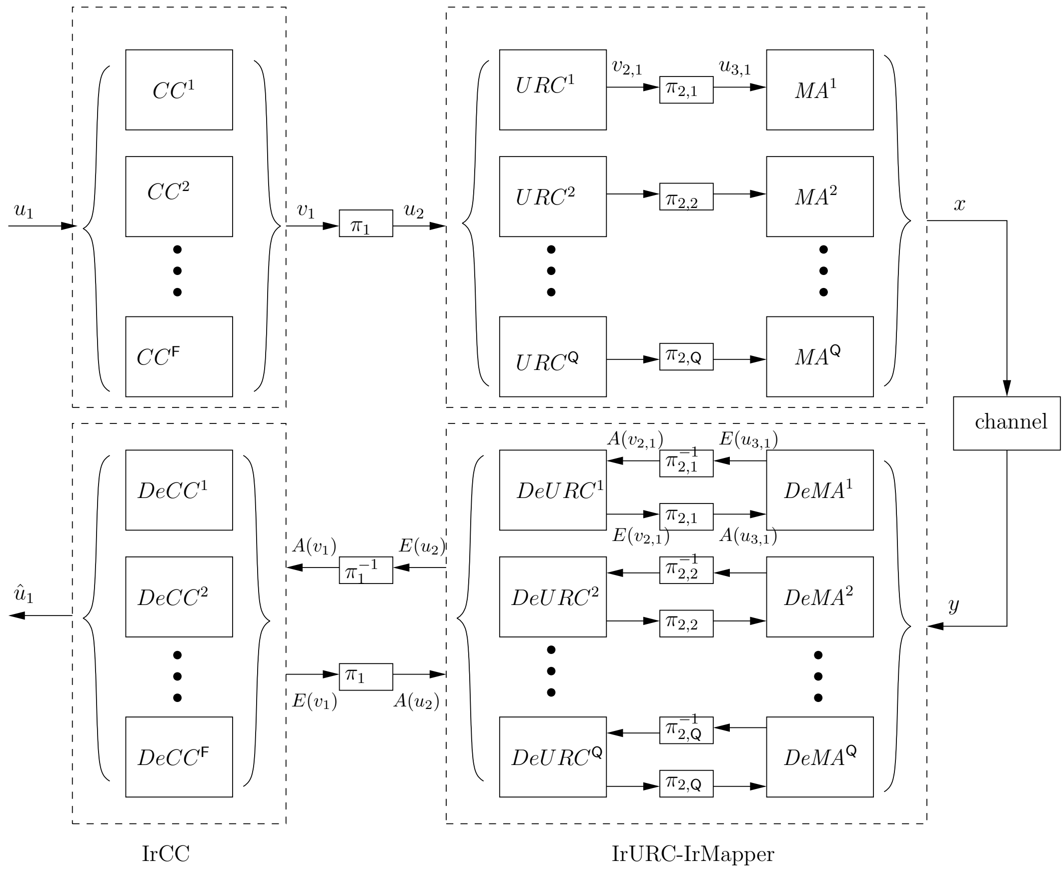


Figure 18.3: The Ir-BICM-ID structure using irregular components, detailing individual subcodes.

18.2.1 System Overview

Each component of the irregular scheme proposed consists of different subcodes. To provide more detailed illustration, the schematic of the proposed Ir-BICM-ID scheme is shown in Figure 18.3, which is constituted by the three main component blocks, namely the IrCC, the bit interleaver and the mapper. In our Ir-BICM-ID design, the IrCC is constituted by F number of low-complexity Convolutional Codes (CC). The inner component module of Figure 18.3 contains Q number of URCs and Q number of MAppers (MA).

The binary source bit stream u_1 is encoded by the IrCC encoder and the encoded bits v_1 are interleaved by the bit interleaver π_1, yielding the permuted bits u_2. The bit stream u_2 is then fed into the IrURC encoder and each of the URC^i codes yields the encoded bits $v_{2,i}$ according to the appropriately assigned weighting coefficients, where i denotes the component index. The resultant bits $v_{2,i}$ are interleaved by the corresponding bit interleaver $\pi_{2,i}$ and the permuted bits $u_{3,i}$ are then mapped to the input of the associated mapper MA^i, as shown in Figure 18.3. This adaptive mapping scheme is referred to as the Irregular Mapper (IrMapper). The modulated symbols x are transmitted via a Rayleigh fading channel and the received symbols y are demodulated.

At the receiver, an iterative decoder is invoked, exchanging extrinsic information between the inner and outer components. The inner iterations exchange extrinsic information between the irregular demapper and the IrURC decoder, where the notation $A(.)$ represents the *a priori* information quantified in terms of LLRs, while $E(.)$ denotes the *extrinsic* information also expressed in terms of LLRs. Observe in Figure 18.3 that each component of the URC decoder and demapper is termed as $DeURC^i$ and $DeMA^i$, respectively and only the first component is labeled for clarity. The inner component is represented by the IrURC-IrMapper block.

By contrast, the outer iterations are invoked between the IrURC decoder and the IrCC decoder of Figure 18.3. Here, each of the IrCC component decoder is termed as $DeCC^i$. Since both the IrURC and IrCC schemes are trellis-based, their decoders employ the MAP algorithm.

The original three-stage arrangement of the IrCC, IrURC and IrMapper schemes would require a 3-D EXIT chart analysis, which is hard to interpret visually. However, our investigations not included here suggest that the inner $DeURC^i$ and $DeMA^i$ blocks require only a few iterations in our 3-stage system to reach the (1,1) point of perfect convergence. We can therefore view these two blocks as a combined decoder using several inner iterations and this allows us to simplify the 3-D EXIT analysis to 2-D EXIT functions, while using the most beneficial activation order of the iterative decoder components.

1) The inner iterative decoder continues iterations, until no more mutual information improvement is achieved.
2) This is then followed by an outer iterative decoding.
3) With the aid of the resultant $a\ priori$ information, the inner iterative decoder is invoked again.
4) This process continues, until the affordable number of iterations is reached.

The resultant inner decoder block of the Ir-BICM-ID scheme is portrayed in Figure 18.4. The deinterleaver within the inner code is denoted as $\pi_{2,i}^{-1}$, where i is the index of the subcode. The corresponding interleaver length of subcode i is constrained to be shorter than the outer interleaver length of π_1, although it would be more beneficial to have a longer interleaver for the sake of achieving a better inner iteration gain.

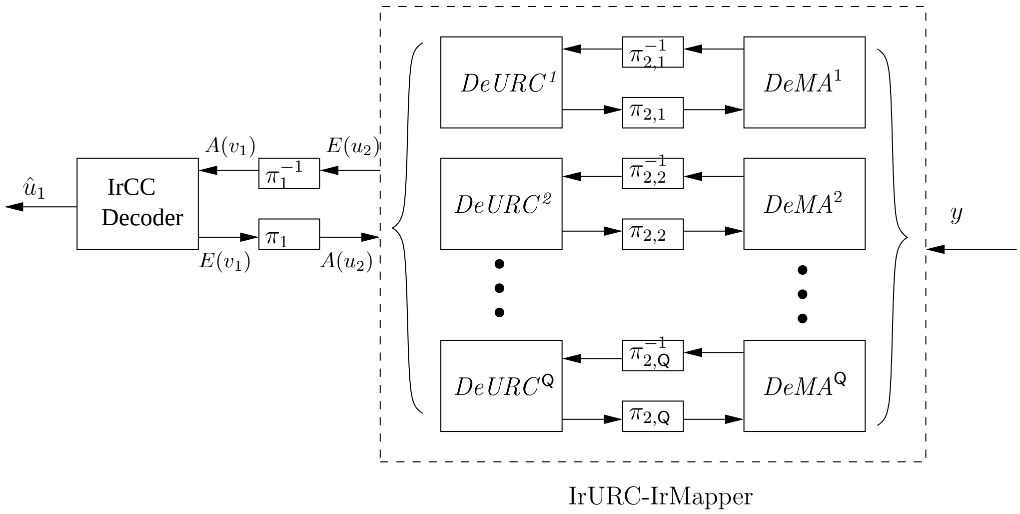


Figure 18.4: The Ir-BICM-ID inner block.

18.3 EXIT Chart Analysis

In this section EXIT chart analysis is performed in order to characterise the scheme's achievable iterative decoding convergence. The basics of EXIT chart analysis were outlined in Section 15.3. Again, the Ir-BICM-ID scheme of Figure 18.3 is a three-stage arrangement, which requires 3-D EXIT analysis, as presented in [371]. However, since the Irregular Demapper (IrDemapper) of Figure 18.4 constitutes a low-complexity component, the low-complexity inner iterations are continued until the mutual information $E(v_{2,i})$ becomes near-constant, because no more extrinsic information may be gleaned by any of these two components, without the third component's intervention. Hence, we can simplify the three-stage EXIT chart representation to a 2-D EXIT curve, as detailed below.

Let $I_{A(b)}$ represent the mutual information between the $a\ priori$ information $A(b)$ and the bit b, while $I_{E(b)}$ denotes the mutual information between the $extrinsic$ information $E(b)$ and the bit b. Observe from Figure 18.3 that the EXIT function of the IrDemapper can be represented as a function of $I_{E(u_3)}$ and the channel's E_b/N_0 value as follows

$$I_{E(u_3)} \quad = \quad \frac{1}{\mathsf{Q}} \sum_{i=1}^{\mathsf{Q}} f_{u_3}[I_{A(u_{3,i})}, E_b/N_0]. \tag{18.1}$$

Observe in Figure 18.3 that in a typical three-stage concatenated design, the IrURC decoder has two mutual information outputs, namely $I_{E(u_2)}$ and $I_{E(v_2)}$, where $I_{E(v_2)}$ is the total mutual information of $I_{E(v_{2,i})}$, $\forall i$. Both of these mutual information components depend on two $a\ priori$ mutual information inputs, namely on $I_{A(u_2)}$ and $I_{A(v_2)}$. The two functions can be defined as

$$I_{E(u_2)} \quad = \quad f_{u_2}[I_{A(u_2)}, I_{A(v_2)}], \tag{18.2}$$

$$I_{E(v_2)} \quad = \quad f_{v_2}[I_{A(u_2)}, I_{A(v_2)}]. \tag{18.3}$$

Again, in this Ir-BICM-ID scheme, we continue invoking inner iterations, until we succeed in generating sufficiently reliable mutual information $I_{E(u_2)}$, before activating the outer iterations. Hence we may combine the inner component blocks according to Figure 18.4, where the dotted box indicates the inner IrURC-IrMapper component. The EXIT function of this inner component can be defined by

$$I_{E(u_2)} = f_{u_2}[I_{A(u_2)}, E_b/N_0]. \tag{18.4}$$

For the IrCC decoder, the EXIT function becomes

$$I_{E(v_1)} = f_{v_1}[I_{A(v_1)}]. \tag{18.5}$$

18.3.1 Area Property

The so-called area-property of EXIT charts [423] can be exploited for creating a near-capacity Ir-BICM-ID scheme based on EXIT curve matching. The area property of EXIT charts [423] states that the area under the EXIT curve of an inner decoder component is approximately equal to the attainable channel capacity, provided that the channel's input symbols are equiprobable. As for the outer component, the area under its EXIT function is equivalent to $(1 - R_o)$, where R_o is the outer component's coding rate. The area properties were formally shown to hold for the Binary Erasure Channel (BEC) [424], but there is experimental evidence that they also holds for AWGN [371] and Rayleigh fading channels [425].

Let A_{v_1} and $\bar{A}_{v_1}$ be the areas under the EXIT function of $f_{v_1}(i)$ and $f_{v_1}^{-1}(i)$, where $i \in [0, 1]$, which can be defined as

$$A_{v_1} = \int_0^1 f_{v_1}(i)di, \quad \bar{A}_{v_1} = \int_0^1 f_{v_1}^{-1}(i)di = 1 - A_{v_1}. \tag{18.6}$$

Therefore the area $\bar{A}_{v_1}$ under the inverse of the EXIT function is approximately equivalent to the coding rate, $\bar{A}_{v_1} \approx R_o$. Since the IrURC has a coding rate of unity, the area A_{v_2} under the combined inner component block's EXIT curve in Figure 18.3, can be defined as follows

$$A_{v_2} \approx C_{channel}, \tag{18.7}$$

where $C_{channel}$ is the achievable channel capacity in the presence of equi-probable symbols. If the inner code rate is not equal to one or the precoder does not have a unity code rate, there would be a capacity loss, which the outer code is unable to recover. Our aim is to create a near-capacity design associated with a narrow EXIT tunnel between the inner and outer EXIT function, reaching the convergence point (1,1), which typically yields an infinitesimally low BER.

18.4 Irregular Components

The irregular components consist of several subcodes, each of which exhibits a different EXIT function, at a different coding rate. The irregular components employed by the Ir-BICM-ID scheme are the IrCC, IrURC and the IrMapper of Figure 18.3, where the IrCC is invoked as the irregular outer component, while the IrURC and IrMapper are combined as the irregular inner component.

18.4.1 Irregular Outer Codes

In this chapter, we introduce a hybrid combination of irregular repetition codes and convolutional codes, which constitute the Ir-BICM-ID encoder. The original outer IrCC component was proposed by Tüchler and Hagenauer [422], which consists of different-rate components created from a mother code by puncturing. To be more specific, the IrCC was designed from a 1/2-rate memory-4 mother code defined by the Generator Polynomials (GPs) $(1, G_1/G_2)$, where puncturing was employed to generate a set of codes having gradually increasing coding rates. The feedback polynomial is defined by $G_1 = 1 + D + D^4 = (31)_8$ and the feedforward polynomial is represented by $G_2 = 1 + D^2 + D^3 + D^4 = (27)_8$.

For lower code rates, two additional generator polynomials, namely $G_3 = 1 + D + D^2 + D^4 = (35)_8$ and $G_4 = 1 + D + D^3 + D^4 = (33)_8$ are employed. The total number of subcodes in the memory-4 IrCC arrangement is $\mathsf{F} = 17$, having code rates spanning the range of [0.1,0.9], with a step size of 0.05.

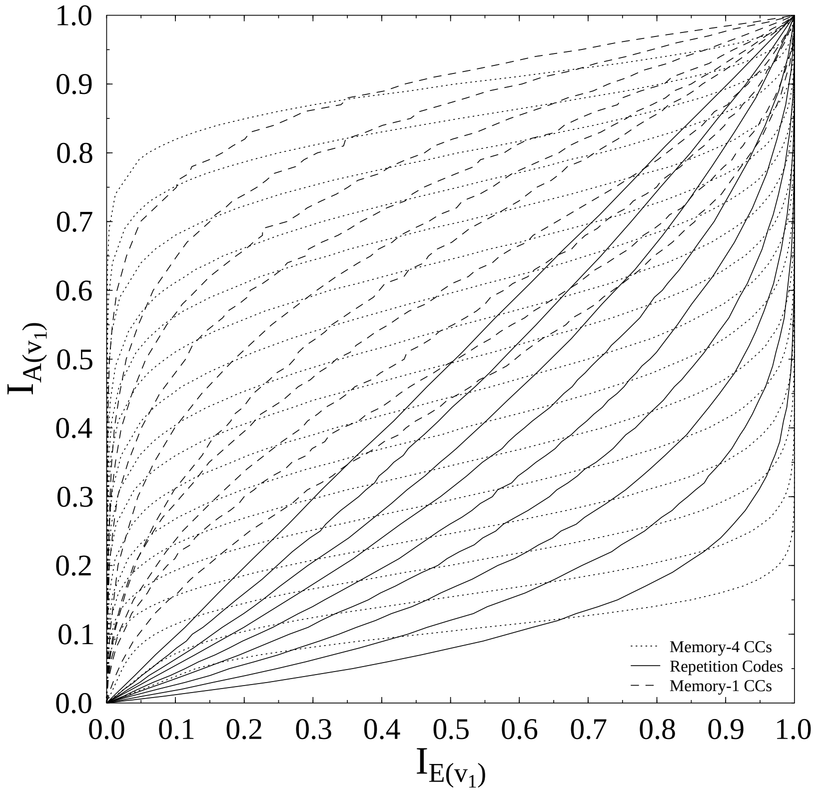


Figure 18.5: EXIT functions of the $\mathsf{F} = 36$ subcodes of the IrCC scheme of Figure 18.3 as specified in Table 18.2.

The EXIT functions of these IrCC subcodes are shown in Figure 18.5, indicated by the dotted lines. Observe in Figure 18.5 that the original memory-4 IrCC exhibits a near-horizontal portion in the EXIT chart, which is typical of strong CCs having a memory of four and associated with 16 trellis states. In order to match the shape of the inner codes' EXIT curves more accurately, the shape of the outer codes' EXIT functions can be adjusted in a way, which allows us to match a more diverse-shaped set of inner-code EXIT functions. Hence we introduce a more diverse range of EXIT functions, particularly along the diagonal region of the EXIT chart. This can be achieved by invoking weaker codes having a lower memory.

Accordingly, memory-1 CCs are also in corporated in the IrCC scheme, which have a simple two-state trellis diagram. The generator polynomial of this memory-1 0.5-rate mother code is defined by GPs $(1, G_1/G_2)$, where we have $G_1 = D$ and $G_2 = 1$. For lower code rates, an extra output GP, namely G_3 is used, where $G_3 = G_2$. For higher code rate, the puncturing pattern of the memory-4 IrCC is employed [422]. This way we generate 10 additional EXIT functions, as shown by the dashed lines in Figure 18.5, spanning the range of [0.45,0.9], with a step size of 0.05. The 10 memory-1 CC are specified by the tuples $\{r, (w_1, w_2, w_3), l, (p_1, p_2, p_3, p_4)\}$, where r represents the code rate, w_i specifies how often the GP G_i occurs, l is the puncturing period and the puncturing pattern is defined by p_i, which is the octal representation associated with G_i [421, 422]. These additional 10 memory-1 CC is detailed in Table 18.1.

r	w_1, w_2, w_3	l	p_1, p_2, p_3, p_4
0.45	1,1,1	9	777,777,021
0.5	1,1	1	1,1
0.55	1,1	11	3777,2737
0.6	1,1	3	7,3
0.65	1,1	13	17777,05253
0.7	1,1	7	177, 025
0.75	1,1	3	7,1
0.8	1,1	4	17,1
0.85	1,1	17	377, 777, 010, 101
0.9	1,1	9	777,1

Table 18.1: The additional 10 memory-1 CC specified by the tuples $\{r, (w_1, w_2, w_3), l, (p_1, p_2, p_3, p_4)\}$.

A repetition code is a simple memoryless code, which consists of only two codewords, namely the all-zero word and the all-one word. Since it has no memory, the EXIT functions of such repetition codes are diagonally-shaped. In Figure 18.5, we have nine different-rate repetition codes as indicated by the solid lines and spanning the code-rate range of [0.1,0.5] with the step size of 0.05.

Each of these $\mathsf{F} = 36$ subcodes encodes a specific fraction of the bit stream u_1 of Figure 18.3 according to a specific weighting coefficient α_i, where $i = 1, 2, ..., 36$. More specifically, let us assume that there are L number of encoded bits v_1 in Figure 18.3, where each subcode i encodes a fraction of $\alpha_i r_i L$ and generates $\alpha_i L$ encoded bits using a coding rate of r_i. Let us assume that there are F number of subcodes and that the following conditions must be satisfied:

$$\sum_{i=1}^{\mathsf{F}} \alpha_i \quad = \quad 1, \tag{18.8}$$

where $R_o = \sum_{i=1}^{\mathsf{F}} \alpha_i r_i$ with $\alpha_i \in [0, 1], \ \forall i$ and R_o is the average outer code rate. The EXIT functions of all $\mathsf{F} = 36$ IrCC subcodes are shown in Figure 18.5. The subcodes of the outer component code are summarised in Table 18.2.

18.4.2 Irregular Inner Codes

In order to generate a narrow but nonetheless open EXIT chart tunnel, which leads to the convergence point of $(I_A, I_E) = (1, 1)$, we have to design inner EXIT functions which match the shape of those in Figure 18.5 and exhibit a wide variety of EXIT characteristic shapes. Again, in this chapter, we propose an inner decoder block, which consists of an IrURC and IrMapper combination.

Code Index i	Type	Coding Rate	Code Index i	Type	Coding Rate
1	CC mem-4	0.10	19	Rep. Code	0.15
2	CC mem-4	0.15	20	Rep. Code	0.20
3	CC mem-4	0.20	21	Rep. Code	0.25
4	CC mem-4	0.25	22	Rep. Code	0.30
5	CC mem-4	0.30	23	Rep. Code	0.35
6	CC mem-4	0.35	24	Rep. Code	0.40
7	CC mem-4	0.40	25	Rep. Code	0.45
8	CC mem-4	0.45	26	Rep. Code	0.50
9	CC mem-4	0.50	27	CC mem-1	0.45
10	CC mem-4	0.55	28	CC mem-1	0.50
11	CC mem-4	0.60	29	CC mem-1	0.55
12	CC mem-4	0.65	30	CC mem-1	0.60
13	CC mem-4	0.70	31	CC mem-1	0.65
14	CC mem-4	0.75	32	CC mem-1	0.70
15	CC mem-4	0.80	33	CC mem-1	0.75
16	CC mem-4	0.85	34	CC mem-1	0.80
17	CC mem-4	0.90	35	CC mem-1	0.85
18	Rep. Code	0.10	36	CC mem-1	0.90

Table 18.2: The outer CC with different number of memories (mem.) and repetition (rep.) subcodes used in the hybrid IrCC component of Figure 18.5, where each subcode index i corresponds to the α_i weighting coefficient in Equation (18.8).

First, we investigate the effect of having different precoder memories, when communicating over an uncorrelated Rayleigh fading channel using 8PSK modulation employing GM. The serial concatenated precoder and mapper represents the inner component block shown in Figure 18.4, although, without inner iterations. We conducted a full search for all URCs, invoking a memory of up to three. The 10 most distinct EXIT functions are plotted in Figure 18.7. The GPs of the URC are defined in the form of (G_1, G_2), where G_1 and G_2 represent the feedforward and feedback polynomials in octal format. The corresponding shift-register component schematics are shown in Figure 18.6.

URC index i	G_1	G_2	URC index i	G_1	G_2
URC_1	2_8	3_8	URC_6	10_8	13_8
URC_2	4_8	7_8	URC_7	15_8	14_8
URC_3	16_8	17_8	URC_8	10_8	17_8
URC_4	7_8	5_8	URC_9	13_8	17_8
URC_5	6_8	7_8	URC_{10}	7_8	6_8

Table 18.3: The outer URC subcodes used in the hybrid IrCC component, where each subcode index i corresponds to the α_i weighting coefficient in Equation (18.8).

Note from Figure 18.7 that the bit-to-8PSK symbol mapper employing GM does not benefit from the iterations, since its EXIT function is a horizontal line, as shown in Figure 15.15. In contrast, the modulator invoking other mappers would require iterations within the inner component, since the corresponding EXIT function is not a horizontal line. Therefore, a near-capacity scheme requires inner iterations between the URC decoder and the demapper of Figure 18.4.

Let us now create a range of IrURC schemes consisting of three URCs, each having a different memory length. After evaluating the EXIT chart of all possible combinations of up to three different-memory URCs, we selected the three most dissimilar URC EXIT functions. More specifically, we used the GPs (G_1, G_2)

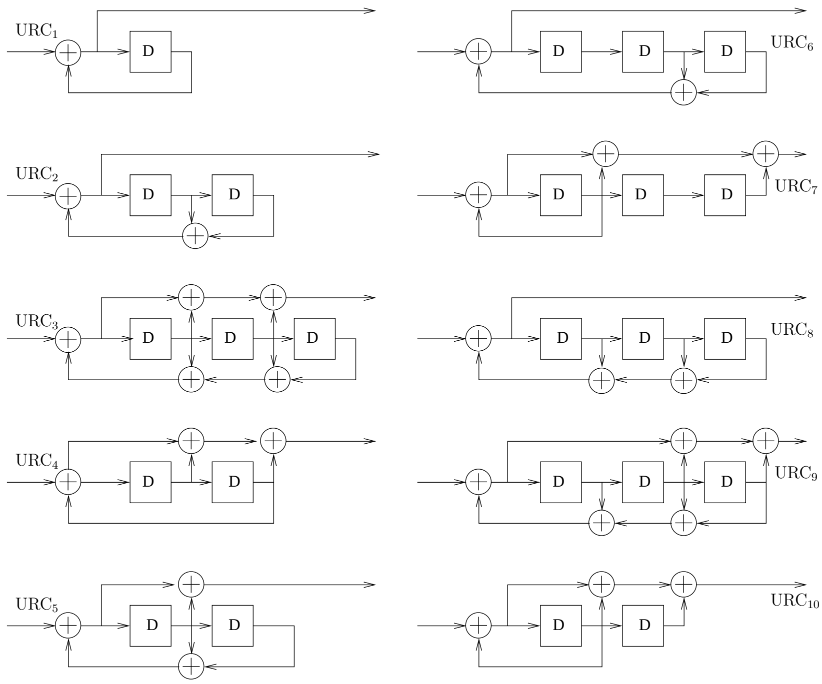


Figure 18.6: Shift-register encoder schematics for $\mathsf{Q} = 10$ URC components.

of $(2,3)_8$, $(4,7)_8$ and $(16,17)_8$, or $(1, 1+D)$, $(1, 1+D+D^2)$ and $(1+D+D^2, 1+D+D^2+D^3)$, which are termed as URC_1 URC_2 URC_3, respectively, as part of the subcodes specified in Table 18.3.

Finally, the IrMapper of Figure 18.3 consists of irregular mapping schemes, each invoking a different bit-to-symbol mapping strategy. Here, we consider an 8PSK constellation and employ four different mapping schemes, which exhibit dissimilar EXIT functions, namely Gray Mapping (GM), Ungerböck's Partitioning (UP) [83], Block Partitioning (BP) and Mixed Partitioning (MP) [334], as defined and specified earlier in Table 15.1. The corresponding bit-to-symbol mapping schemes are repeated in Table 18.4 for convenience.

With the IrURC and IrMapper schemes defined, we proceed by creating $\mathsf{Q} = 12$ different EXIT functions for the inner decoder components, each invoking a different combination of the IrURC and IrMapper schemes of Table 18.4. For example, the URC_1 scheme employing the GM arrangement was defined in Table 18.4 as UM_1.

The EXIT functions of the $\mathsf{Q} = 12$ combined inner IrURC-IrMapper components are plotted in Figure 18.8 for $E_b/N_0 = 5.3$dB. Note that these EXIT functions exhibit a wider range of shapes compared to the non-iterative, Gray-mapped IrURC shown in Figure 18.7. The weighting coefficients are defined as β, satisfying the following conditions:

$$\sum_{i=1}^{\mathsf{Q}} \beta_i = 1 \quad \text{and } \beta_i \in [0,1], \ \ \forall i. \tag{18.9}$$

18.4.3 EXIT Chart Matching

We adopt the EXIT chart matching algorithm of [422] to jointly match the EXIT functions of both the irregular inner and outer components, as detailed in Section 18.4.2. The EXIT functions to be considered are described in Equations (18.4) and (18.5). More explicitly, we intend to minimise the EXIT-tunnel area

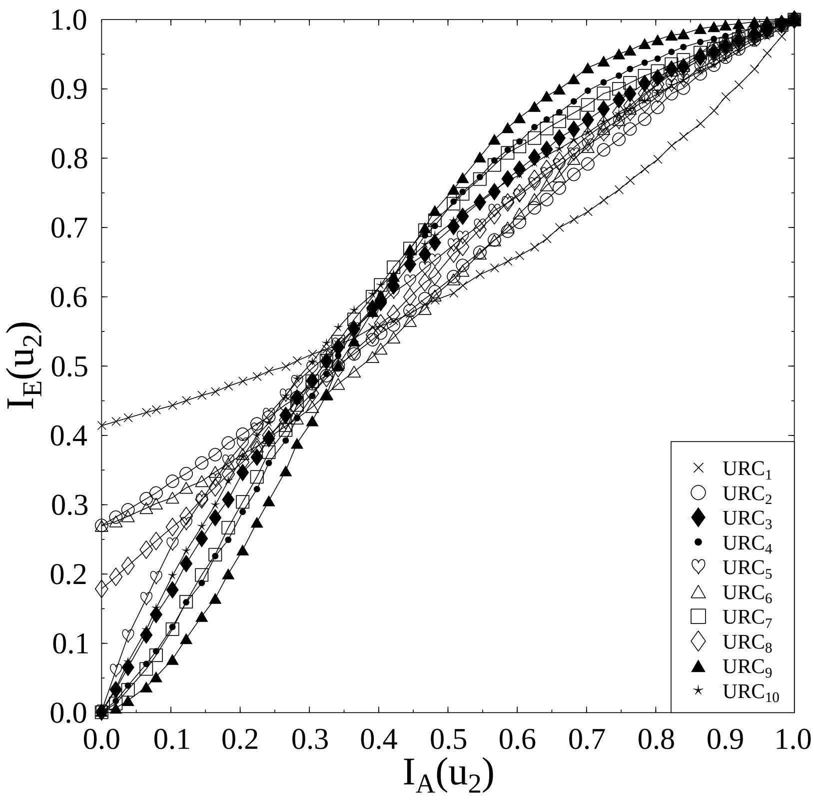


Figure 18.7: EXIT functions of the $\mathsf{Q} = 10$ inner URC sub-components of Figure 18.6, specified in Table 18.3, when communicating over uncorrelated Rayleigh fading channels at $E_b/N_0 = 5.3$dB. The precoders of Table 18.3 having memories of 1, 2 and 3 were employed in conjunction with an 8PSK GM using no inner iteration.

represented by the square of the error function of

$$e(i) = [f_{v_2}(i, E_b/N_0) - f_{v_1}^{-1}(i)]. \tag{18.10}$$

Furthermore, the error function should be larger than zero and may be expressed as [422]:

$$\begin{aligned} \mathcal{J}(\alpha_1, ..., \alpha_{\mathsf{F}}) &= \int_0^1 e(i)^2 di, \;\; e(i) > 0, \forall i \in [0,1], \quad \text{OR} \\ \mathcal{J}(\beta_1, ..., \beta_{\mathsf{Q}}) &= \int_0^1 e(i)^2 di, \;\; e(i) > 0, \forall i \in [0,1], \end{aligned} \tag{18.11}$$

where Q or F is the number of irregular sub-codes used either by the inner or by the outer components, depending on where the matching process is executed. We term the constraint of Equation (18.11) as condition C:1. Another constraint we impose here is that of ensuring that Equations (18.8) and (18.9) are fulfilled and we term these as condition C:2 and C:3, respectively. The term function $\mathcal{J}(.)$ of Equation (18.11) satisfies these conditions, depending on where the matching process is executed. When the matching algorithm is executed with respect to the outer codes, the term α_i of Equation (18.8) would be used and when it is executed with respect to the inner codes, the term β_i of Equation (18.9) would be considered.

The joint EXIT chart matching algorithm of [422] is applied alternatively to the inner and outer components in order to find the optimal value of $\alpha_{\mathbf{opt}}$ and $\beta_{\mathbf{opt}}$ which is summarised as follows

Mapping Type	Mapping Indices to Corresponding Signal Points (cos$2\pi i/M$, sin$2\pi i/M$) for $i \in$ 0 1 2 3 4 5 6 7
GM	0 1 3 2 6 7 5 4
UP	0 1 2 3 4 5 6 7
BP	7 3 6 2 4 0 5 1
MP	0 2 1 7 4 6 5 3
Inner Component	URC Type/Mapping Type
UM_1	URC_1/GM
UM_2	URC_2/GM
UM_3	URC_3/GM
UM_4	URC_1/UP
UM_5	URC_2/UP
UM_6	URC_3/UP
UM_7	URC_1/BP
UM_8	URC_2/BP
UM_9	URC_3/BP
UM_{10}	URC_1/MP
UM_{11}	URC_2/MP
UM_{12}	URC_3/MP

Table 18.4: Various mapping schemes and URC Mapper (UM) combinations, each exhibiting a different inner EXIT function. The bit-to-symbol mapping strategies were Gray Mapping (GM), Ungerböck's Partitioning (UP), Block Partitioning (BP) and Mixed Partitioning (MP) [334], where M is the number of constellation points.

Step 1: Create the F outer components of the IrCC and the Q inner URC Mapper components (UM), as shown in Figures 18.5 and 18.7.

Step 2: Select one out of Q UMs, as the inner component to be used.

Step 3: Select an initial outer coding rate R_0, to be employed in the EXIT-chart matching algorithm of [422]. This R_0 is the initial R_{IR} to be placed in the matrix $\mathbf{d}$ as detailed in [422].

Step 4: Employ the EXIT chart matching algorithm [422] to obtain $\alpha_{\mathbf{opt}}$, subject to the constraints of C:1 and C:2, given by Equations (18.11) and (18.8).

Step 5: Repeat $\mathbf{Step\ 3}$ and $\mathbf{Step\ 4}$ iteratively, until a sufficiently high initial rate R_0 is obtained.

Step 6: Record the resultant outer EXIT curve.

Step 7: Invoke the EXIT chart matching algorithm for finding the best weights of the Q number of UMs to match the the IrCC's outer EXIT curve of $\mathbf{Step\ 6}$, in order to obtain $\beta_{\mathbf{opt}}$. This process is subject to the constraints of C:1 and C:3, given by Equations (18.11) and (18.9).

Step 8: Record the resultant inner EXIT curve and repeat the EXIT chart matching process of $\mathbf{Step\ 4}$, $\mathbf{Step\ 6}$ and $\mathbf{Step\ 7}$, each time with a small increment of R_0, until no more increment is possible.

Step 9: Activate the EXIT matching algorithm to find the best-matching outer code EXIT chart function for the target inner code EXIT function, this time using one out of Q UMs, which was not used in $\mathbf{Step\ 2}$. Repeat $\mathbf{Step\ 4}$, $\mathbf{Step\ 6}$, $\mathbf{Step\ 7}$ and $\mathbf{Step\ 8}$, until all the Q number of UMs were tested.

Step 10: Terminate the algorithm, choose the best values of $\alpha_{\mathbf{opt}}$ and $\beta_{\mathbf{opt}}$, yielding the highest possible outer coding rate.

Eventually, at the end of the EXIT chart matching process, the EXIT function $f_{v_1}[I_{A(v_1)}]$ of Equation (18.5) corresponds to the best IrCC, which might be constructed from $\mathsf{F} = 36$ subcodes, where each

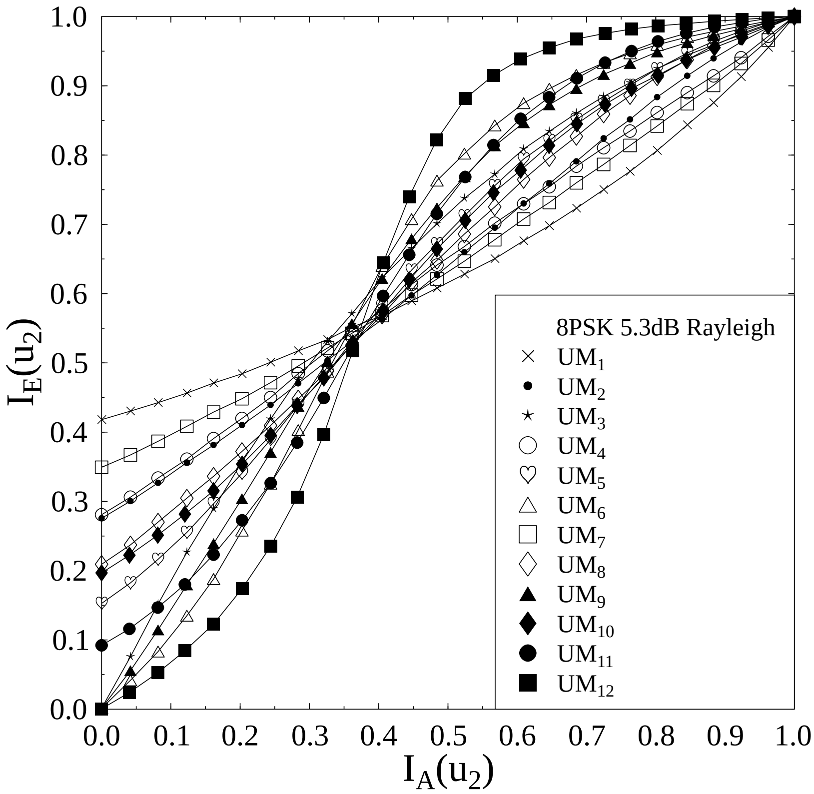


Figure 18.8: EXIT functions of the $\mathsf{Q} = 12$ inner sub-components of Figure 18.4 and Table 18.4, when communicating over uncorrelated Rayleigh fading channel at $E_b/N_0 = 5.3$dB.

subcode EXIT function is defined as $f^i_{v_1}[I_{A(v_1)}]$. This also implies that the EXIT function $f_{v_1}[I_{A(v_1)}]$ is the weighted superposition of $\mathsf{F} = 36$ EXIT functions, satisfying the following conditions

$$f_{v_1}[I_{A(v_1)}] = \sum_{i=1}^{\mathsf{F}} \alpha_i f^i_{v_1}[I_{A(v_1)}]. \tag{18.12}$$

In a similar fashion, the EXIT function of $f_{u_2}[I_{A(u_2)}, E_b/N_0]$ in Equation (18.4) corresponds to that specific irregular inner component, which might be constructed from $\mathsf{Q} = 12$ subcomponents. The weighted superposition of the EXIT function $f^i_{u_2}[I_{A(u_2)}, E_b/N_0]$ of each sub-component results in

$$f_{u_2}[I_{A(u_2)}, E_b/N_0] = \sum_{i=1}^{\mathsf{Q}} \beta_i f^i_{u_2}[I_{A(u_2)}, E_b/N_0]. \tag{18.13}$$

18.5 Simulation Results

In this section we embark on characterising the proposed Ir-BICM-ID scheme in terms of its EXIT chart based convergence behaviour for transmission over the uncorrelated Rayleigh fading channel. Let us employ the EXIT chart matching algorithm described in Section 18.4.3, invoking the hybrid combination the IrCC, IrURC as well as IrMapper schemes. The adjustable shapes of the EXIT functions enable us to reduce the open EXIT tunnel area and hence to create a near-capacity Ir-BICM-ID scheme. As a further benefit, we are able to shift the inner EXIT function closer to the $(I_A, I_E) = (1, 1)$ point for the sake of achieving an infinitesimally low BER.

Since the inner IrURC component has a unity rate by definition, the effective throughput of the Ir-BICM-ID scheme may be expressed as $\eta = R \cdot \log_2 M$ bits per channel use, where M is the number of constellation points and R denotes the coding rate of the outer IrCC. The relationship of the SNR (E_s/N_0) and E_b/N_0 can be represented as $E_s/N_0 = E_b/N_0 \cdot \eta$, where N_0 is the noise power spectral density and E_s as well as E_b denotes the transmit energy per channel symbol use and per bit of source information, respectively.

Later in this section, we will analyse the complexity of the near-capacity IrCC scheme, which can be quantified in terms of the number of trellis states of the IrCC and IrURC components. The complexity of the IrMapper is low and hence it is ignored in the complexity calculation. In order to reduce the complexity, we are also able to adjust the weighting coefficient α_i for the various subcodes CC^i, creating a flexible Ir-BICM-ID scheme.

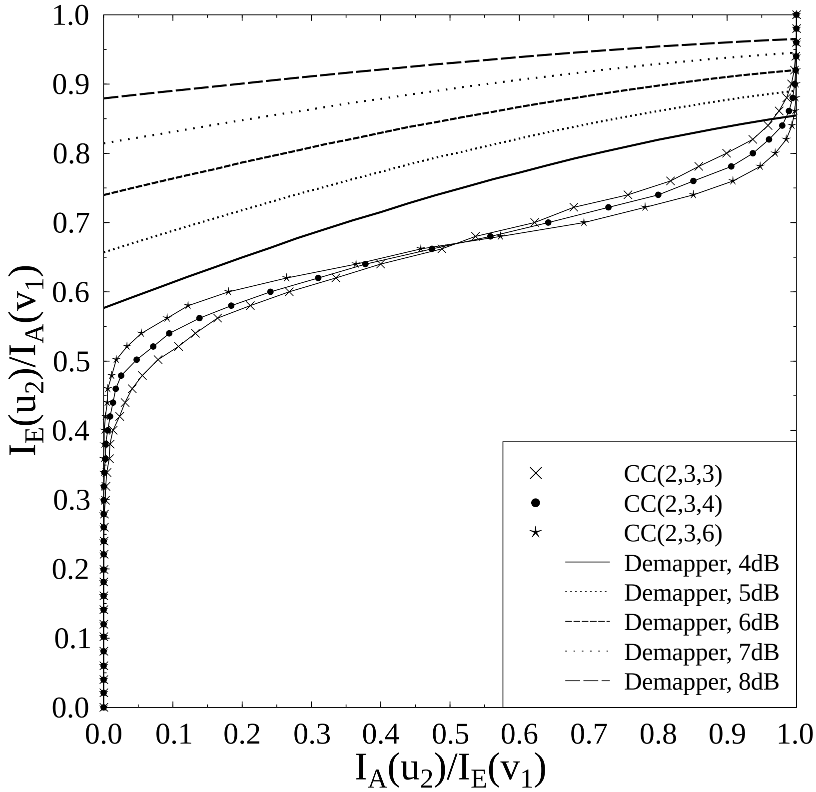


Figure 18.9: EXIT functions of the classic BICM-ID inner and outer components of Figure 18.1, when transmitting over an uncorrelated Rayleigh fading channel using 8PSK UP modulation.

The conventional BICM-ID scheme dispenses with the URC, hence the inner component consists of a simple demapper. Therefore in Figure 18.3, the dashed box surrounding the inner component representing the BICM-ID scheme is constituted solely by the demapper. The outer code, is constituted by a convolutional code. Figure 18.9 illustrates the EXIT functions of both the inner and outer components, where the outer code rate was 2/3, associated with a memory of $\varphi = 3, 4$ and 6.

Observe from Figure 18.9 that the inner demapper does not reach the point of convergence at (1,1).

Furthermore, there is a mismatch between the corresponding EXIT curve shapes, indicating an E_b/N_0 'loss' with respect to the capacity. A further E_b/N_0 improvement is achieved upon introducing the IrCC outer code, which reduces the area of the open EXIT tunnel, as shown in Figure 18.10, but still exhibits a 'larger-than-necessary' EXIT tunnel. Figure 18.10 shows that the shape of the outer IrCC EXIT function is better matched to that of the inner codes, as indicated by the dotted line shifting upwards, when the channel's E_b/N_0 value increases.

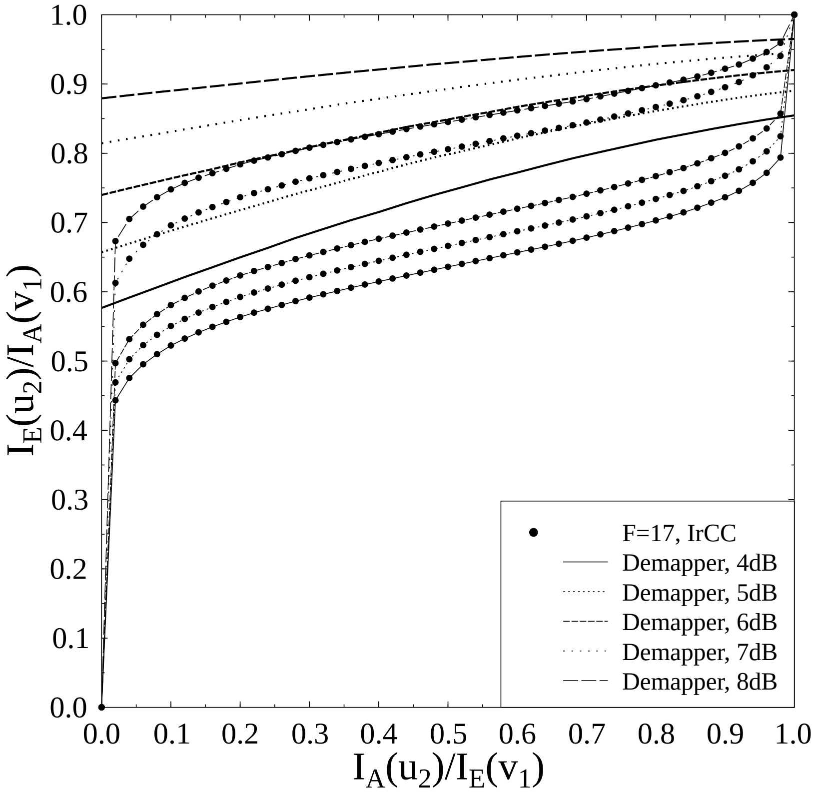


Figure 18.10: EXIT functions of the BICM-ID inner and outer components, using a $\mathsf{F} = 17$ IrCC of memory-4 as the outer component, when communicating over an uncorrelated Rayleigh fading channel invoking 8PSK UP modulation. The general structure of the schematic is shown in Figure 18.1 by replacing the CC encoder with the $\mathsf{F} = 17$ IrCC of memory-4, as detailed in Table 18.2.

Let us finally employ the EXIT chart matching algorithm described in Section 18.4.3, invoking the IrCC, IrURC as well as IrMapper schemes. The shape of the EXIT functions enables us to reduce the open EXIT tunnel area and hence to create a near-capacity Ir-BICM-ID scheme. As a further benefit, we are able to shift the inner EXIT function closer to the (1,1) point for the sake of achieving an infinitesimally low BER.

We observe from Figure 18.11 that the open EXIT tunnel of the resultant scheme is narrow and reaches the point of convergence at $(I_A, I_E) = (1, 1)$. However, since the number of iterations required increases, the decoding complexity is also increased. Figure 18.11 also illustrates that the trajectory matches the inner and outer EXIT functions and evolves within the narrow tunnel, reaching the $(I_A, I_E) = (1, 1)$ point of convergence for an interleaver length of 300,000.

Note that the proposed Ir-BICM-ID scheme has advantages over the bit-interleaved irregular modu-

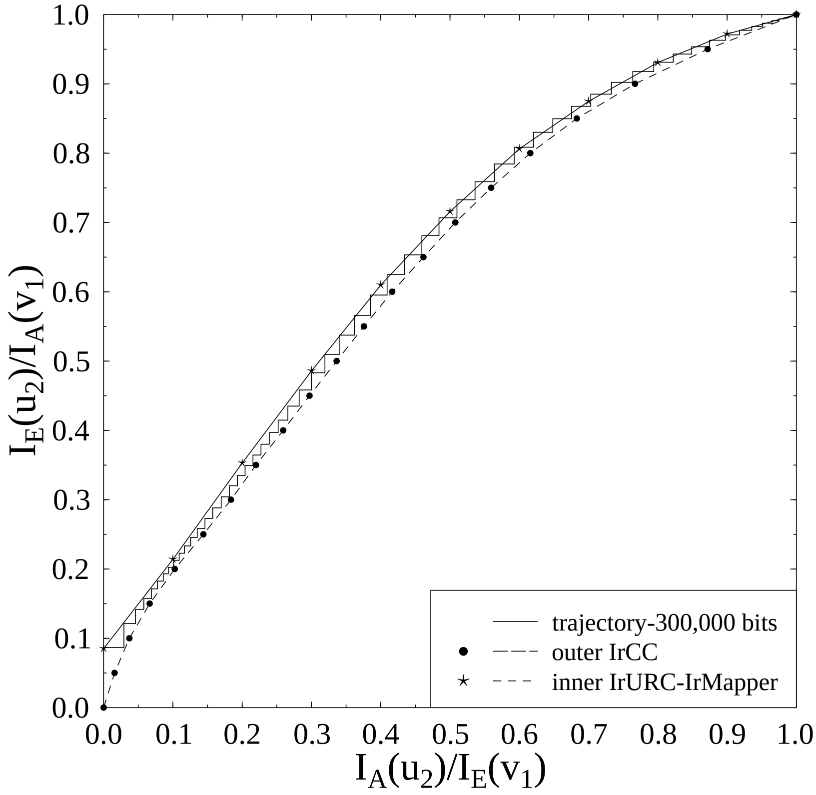


Figure 18.11: EXIT functions and the trajectory of the Ir-BICM-ID schemes of Figure 18.3 designed for transmission over an uncorrelated Rayleigh fading channel at $E_b/N_0 = 5.3$dB. The overall code rate is 0.635, the effective throughput is 1.905 bit/symbol and the modulation scheme is 8PSK. The weighting coefficients $\alpha_{\mathbf{opt}}$ and $\beta_{\mathbf{opt}}$ are given in Equations (18.14) and (18.15), while the individual subcodes are detailed in Tables 18.2 and 18.4.

lation scheme of [417], when we further explore the effects of various mapping schemes combined with URCs having different memory lengths. This gives us the flexibility of adjusting the EXIT curve shape in order to achieve a low BER, without having to change the number of modulated constellation points, which would require complex state-of-the-art AMC [418]. Furthermore, we employ the joint EXIT chart matching algorithm of Section 18.4.3 to produce flexible inner and outer component codes. The complexity imposed by the iterations between the IrURC and IrMapper schemes is low compared to that of the outer IrCC. Our hybrid IrCC scheme employs low-memory CCs and two-state memoryless repetition codes, which allows us to reduce the trellis complexity of the scheme.

The EXIT function of Figure 18.11 was recorded for the EXIT chart matching algorithm of Section 18.4.3 for an overall coding rate of 0.635. The resultant weighting coefficients of $\alpha_{\mathbf{opt}}$ and $\beta_{\mathbf{opt}}$ are

$$\begin{aligned}
\alpha_{\mathbf{opt}} &= [\alpha_1, ..., \alpha_{36}] \\
&= [0, 0, 0.0311597, 0, 0, 0, 0, 0, 0, 0, 0.0293419, 0.0228237, 0, 0.0309682, \\
&\quad 0, 0, 0, 0, 0, 0, 0.0837582, 0, 0, 0, 0.404757, 0, 0, 0, 0, 0, 0, 0, 0, 0.397229] \\
&= [\alpha_3^{0.0311597}, \alpha_{12}^{0.0293419}, \alpha_{13}^{0.0228237}, \alpha_{15}^{0.0309682}, \alpha_{23}^{0.0837582} \\
&\quad \alpha_{27}^{0.404757}, \alpha_{36}^{0.404757}], \qquad (18.14) \\
\beta_{\mathbf{opt}} &= [\beta_1, ..., \beta_{12}] \\
&= [0, 0, 0.340268, 0, 0.550512, 0, 0, 0, 0.109219, 0, 0, 0] \\
&= [\beta_3^{0.340268}, \beta_5^{0.550512}, \beta_9^{0.109219}], \qquad (18.15)
\end{aligned}$$

where α_i^j and β_i^j refer to the weighting coefficient of α_i and β_i having a value of j. Those components having 0 weighting coefficient are ignored in the expression for clarity.

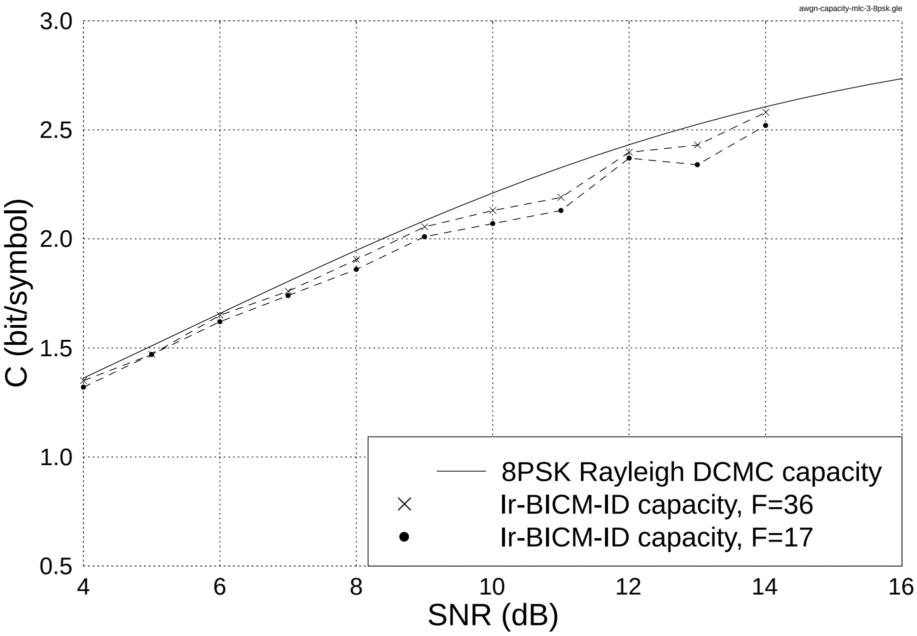


Figure 18.12: The maximum effective throughput of the propose Ir-BICM-ID scheme of Figure 18.3 in comparison to the theoretical DCMC capacity plot [426] using 8PSK-modulated transmission over the uncorrelated Rayleigh fading channel.

The theoretical Discrete-input Continuous-output Memoryless Channel's (DCMC) capacity [390] was described in Section 16.5.2 which is plotted in comparison to the maximum achievable capacity of the proposed Ir-BICM-ID scheme in Figure 18.12. Note that the achieveable capacity of the Ir-BICM-ID scheme is close to the DCMC's capacity. For example, at SNR = 6dB the discrepancy between the theoretical capacity and the proposed coded modulation scheme's is less than 0.1dB in Figure 18.12. This confirms the benefits of the proposed EXIT chart matching approach.

The dashed line combined with the dot marker of Figure 18.12 shows the achievable throughput of the Ir-BICM-ID scheme, when employing Tüchler's $\mathsf{F} = 17$ memory-4 IrCC [422]. Since the EXIT functions of the 17 subcodes consist of memory-4 CCs, the EXIT curves exhibit a rather gently shaping, near-horizontal shapes in their middle section as can been seen in Figure 18.5. Therefore the matching process exhibits a higher discrepancy from the DCMC's capacity. When introducing the whole range of $\mathsf{F} = 36$

subcodes for the new hybrid IrCCs, we attain a closer match to the channel capacity, as shown by the cross-markers of Figure 18.12.

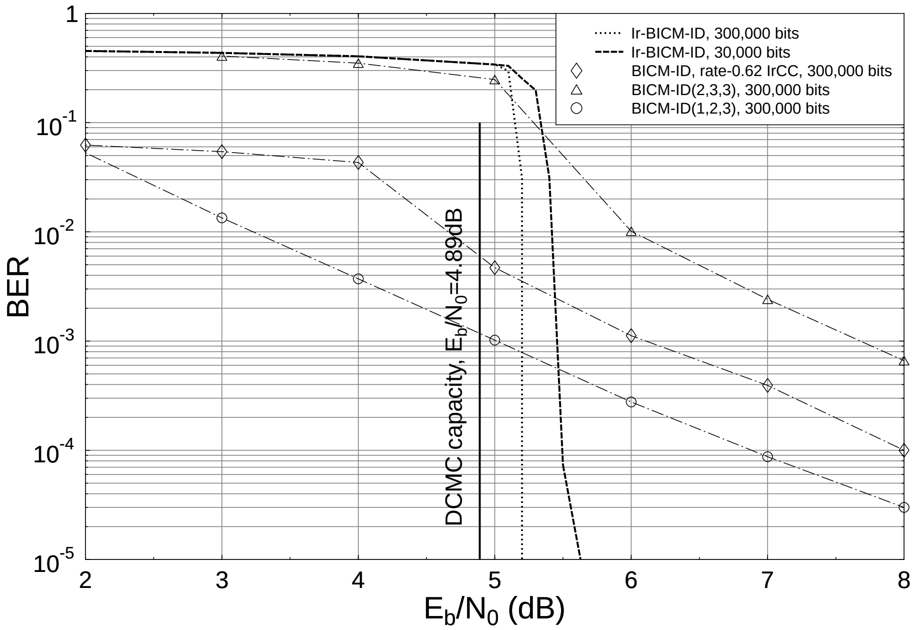


Figure 18.13: The BER performance of rate-0.635 Ir-BICM-ID schemes of Figure 18.3 at an effective throughput of 1.905 bit/symbol, using two different interleaver lengths and that of the classic 8PSK UP BICM-ID of Figure 18.1, employing a coding rate of $R = 1/2$ and $2/3$, respectively. A BER benchmarker of BICM-ID scheme invoking rate-0.62 outer IrCC proposed in [422] employing UP mapper is also attained. All subcodes of the irregular codes invoking IrCC and UM are detailed in Tables 18.2 and 18.4.

Figure 18.13 shows the BER performance of the Ir-BICM-ID scheme for different interleaver lengths. The DCMC's capacity limit is $E_b/N_0 = 4.89$dB at a throughput of 1.905 bit/symbol using 8PSK. By interleaving over 300,000 bits, the Ir-BICM-ID design becomes capable of achieving an infinitesimally low BER as close as 0.32dB from the theoretical capacity limit, as shown in Figure 18.13. When we decrease the interleaver length by a factor of 10 to 30,000 bits, the performance degrades, but still remains within about 0.75dB of the capacity. We include the classic 1/2- and 2/3-rate BICM-ID benchmarker scheme [90] having a 300,000-bit interleaver length as well as the BICM-ID scheme employing the $\mathsf{F} = 17$ IrCC [422] outer code of rate-0.62. All the benchmarkers exhibit BER $> 10^{-5}$ even at $E_b/N_0 = 8$dB, which is relatively far from the capacity limit of $E_b/N_0 = 4.89$dB.

The complexity of the scheme is illustrated in Figure 18.14, quantified in terms of the number of trellis states for both the IrCC and IrURC schemes. As shown in Figure 18.14, the system requires a higher complexity for performing closer to capacity, namely at $E_b/N_0 = 5.3$dB. Note that we involve a total of about 6×10^8 trellis states of the IrURC and IrCC decoder for decoding 300,000 bits. By increasing the E_b/N_0 values, we are able to reduce the complexity of the system, as the EXIT tunnel becomes wider. Figure 18.15 illustrates the benefits of increasing E_b/N_0 from 4dB to 10dB. At $E_b/N_0 = 6.5$dB, the number of trellis states involved decreases exponentially to around 1.8×10^8.

Another way of reducing the complexity is to decrease the IrCC scheme's coding rate, in which case we would also sacrifice the near-capacity performance. The EXIT functions recorded for coding rates of $R = 0.5$ and $R = 0.4$ are shown in Figure 18.16, where the number of outer iterations was I = 10 and 7, respectively. The weighting coefficients α_{opt} for the IrCC having coding rates of $R = 0.5$ and $R = 0.4$ are

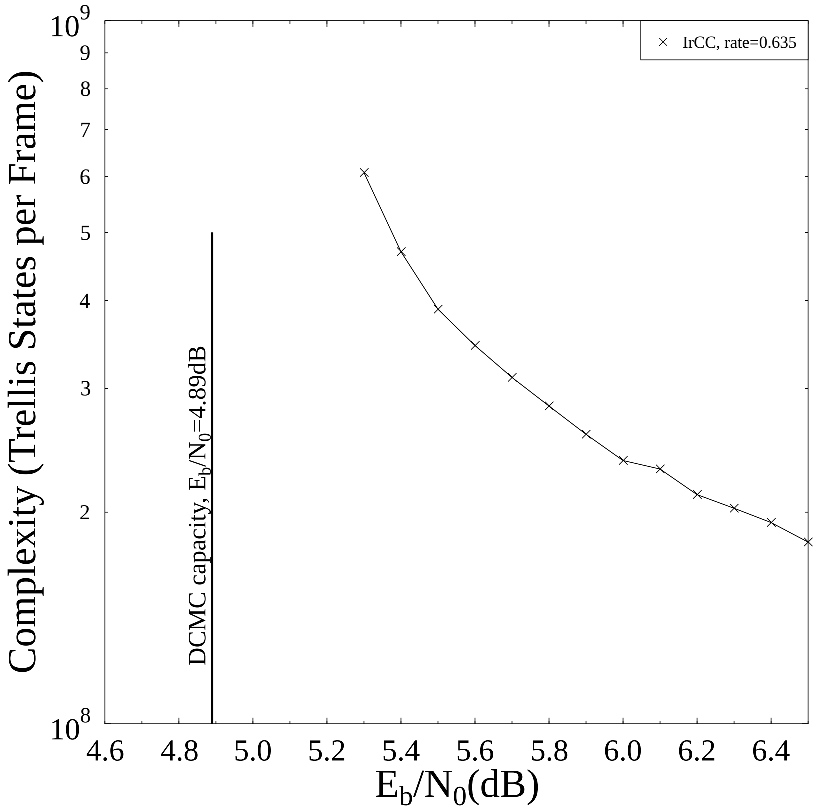


Figure 18.14: The complexity of the Ir-BICM-ID scheme of Figure 18.3 measured with respect to the number of trellis states employing the same weighting coefficients as in Figure 18.13 to achieve a BER of 10^{-5}. This scheme communicated over an uncorrelated Rayleigh fading channel using interleaver over a total of 300,000 bits/frame.

detailed in Table 18.5.

18.6 Chapter Conclusions

In this chapter, we proposed an Ir-BICM-ID scheme, invoking irregular encoders, precoders and modulators. Each component consists of irregular structures of various subcodes. Section 18.2 details the structure of the Ir-BICM-ID scheme specifying the activation order of both the outer and inner components. This three-stage concatenated system can be regarded as a two-stage system, when treating the IrURC and IrMapper as the combined inner bock. The EXIT chart of Figure 18.8 in Section 18.3 illustrates the corresponding 2-D EXIT functions.

The irregular IrCC components are further improved by introducing more EXIT functions with the aid of designing new memory-1 CCs and repetition codes, as described in Section 18.4 of Table 18.2. The outer component of the hybrid IrCC consists of $\mathsf{F} = 36$ subcodes, while the inner component consists of $\mathsf{Q} = 12$ subcodes, which are given in Tables 18.2 and 18.4, respectively. As detailed in Section 18.4.2, the proposed inner code consist of a combination of memory-1 up to memory-3 IrURCs, each invoking various mapping strategies, as outlined in Table 18.4.

The EXIT matching algorithm of [422] was modified to match both the inner and outer irregular components, as described in Section 18.4.3. The result of this matching procedure is the weighted superposition

IrCC Coding Rate	Weighting Coefficient $\alpha_{\mathbf{opt}} = [\alpha_1, ..., \alpha_{36}]$
0.5	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.168897, 0.0151061, 0, 0, 0.158275, 0.124163, 0, 0, 0, 0.533687, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] $[\alpha_{12}^{0.168897}, \alpha_{13}^{0.0151061}, \alpha_{16}^{0.158275}, \alpha_{17}^{0.124163}, \alpha_{21}^{0.533687}]$
0.4	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.157339, 0, 0, 0.139792, 0.0657432, 0, 0.195015, 0, 0, 0.442172, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] $[\alpha_{12}^{0.157339}, \alpha_{15}^{0.139792}, \alpha_{16}^{0.0657432}, \alpha_{18}^{0.195015}, \alpha_{21}^{0.442172}]$

Table 18.5: Weighting coefficients $\alpha_{\mathbf{opt}}$ for IrCC having coding rates of $R = 0.5$ and $R = 0.4$.

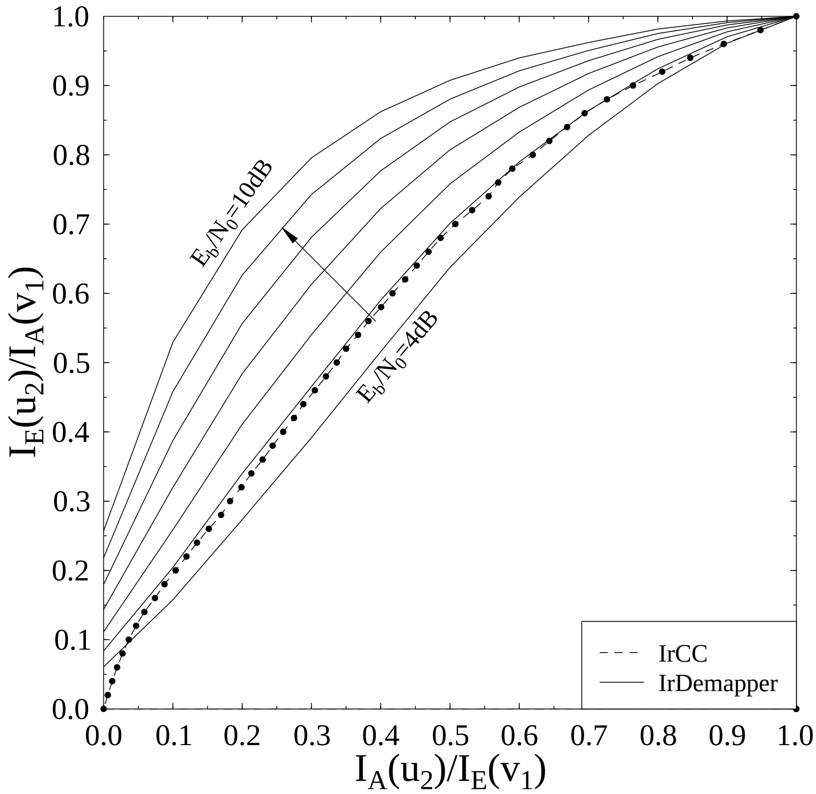


Figure 18.15: The EXIT functions of Ir-BICM-ID scheme of Figure 18.3 for the inner component having a coding rate of $R = 0.635$, when communicating over an uncorrelated Rayleigh fading channel at various E_b/N_0 values. The weighting coefficients $\alpha_{\mathbf{opt}}$ and $\beta_{\mathbf{opt}}$ are detailed in Equations (18.14) and (18.15), respectively.

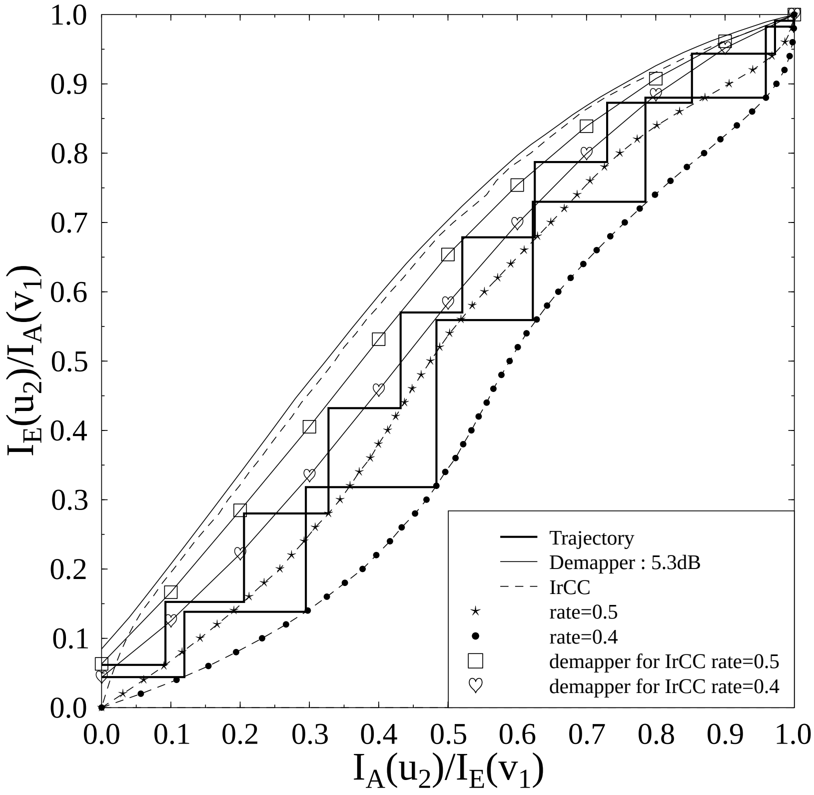


Figure 18.16: The EXIT functions for IrCCs of coding rates $R = 0.5$ and $R = 0.4$, when communicating over an uncorrelated Rayleigh fading channel. The weighting coefficients $\alpha_{\mathbf{opt}}$ and $\beta_{\mathbf{opt}}$ are detailed in Table 18.5 and Equation (18.15), respectively.

of both the inner and outer EXIT functions, having the weighting coefficients of α_{opt} and β_{opt}. This results in a narrow EXIT tunnel, as shown in Figure 18.11.

Section 18.5 illustrated the attainable performance of the proposed Ir-BICM-ID scheme. Figure 18.12 quantifies the discrepancy of this new scheme with respect to the theoretical capacity. At low SNR, for example at SNR = 6dB, the capacity of the system can be as close as 0.1dB from the capacity. The BER curve of the system exhibits an infinitesimally low value at about 0.32dB from the DCMC's capacity limit, when using an interleaver length of 300,000 bits, as shown in Figure 18.13.

Due to its close-to-capacity performance, the Ir-BICM-ID may require a high number of iterations. Figure 18.14 shows the complexity of the scheme quantified in terms of the total number of decoding trellis states of the IrCC and IrURC schemes. We can observe that the closer this system operates to the capacity, the higher the implementational complexity imposed. The complexity decreases exponentially, as the distance from capacity increases. However, the complexity imposed may be reduced by increasing the operational E_b/N_0 values, as shown in Figure 18.15. Furthermore, by adjusting α_{opt} in order to reduce the coding rate, we can increase the width of the EXIT tunnel, as seen in Figure 18.16.

Bibliography

[1] C. E. Shannon, “A mathematical theory of communication,” *Bell System Technical Journal*, pp. 379–427, 1948.

[2] R. Hamming, “Error detecting and error correcting codes,” *Bell System Technical Journal*, vol. 29, pp. 147–160, 1950.

[3] P. Elias, “Coding for noisy channels,” *IRE Convention Record, pt.4*, pp. 37–47, 1955.

[4] J. Wozencraft, “Sequential decoding for reliable communication,” *IRE Natl. Conv. Rec.*, vol. 5, pt.2, pp. 11–25, 1957.

[5] J. Wozencraft and B. Reiffen, *Sequential Decoding*. Cambridge, MA, USA: MIT Press, 1961.

[6] R. Fano, “A heuristic discussion of probabilistic coding,” *IEEE Transactions on Information Theory*, vol. IT-9, pp. 64–74, April 1963.

[7] J. Massey, *Threshold Decoding*. Cambridge, MA, USA: MIT Press, 1963.

[8] A. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Transactions on Information Theory*, vol. IT-13, pp. 260–269, April 1967.

[9] G. Forney, “The Viterbi algorithm,” *Proceedings of the IEEE*, vol. 61, pp. 268–278, March 1973.

[10] J. H. Chen, “High-quality 16 kb/s speech coding with a one-way delay less than 2 ms,” in *Proceedings of International Conference on Acoustics, Speech, and Signal Processing, ICASSP’90*, vol. 1, (Albuquerque, New Mexico, USA), pp. 453–456, IEEE, 3–6 April 1990.

[11] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, “Optimal Decoding of Linear Codes for Minimising Symbol Error Rate,” *IEEE Transactions on Information Theory*, vol. 20, pp. 284–287, March 1974.

[12] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes,” in *Proceedings of the International Conference on Communications*, (Geneva, Switzerland), pp. 1064–1070, May 1993.

[13] C. Berrou and A. Glavieux, “Near optimum error correcting coding and decoding: Turbo codes,” *IEEE Transactions on Communications*, vol. 44, pp. 1261–1271, October 1996.

[14] A. Hocquenghem, “Codes correcteurs d’erreurs,” *Chiffres (Paris)*, vol. 2, pp. 147–156, September 1959.

[15] R. Bose and D. Ray-Chaudhuri, “On a class of error correcting binary group codes,” *Information and Control*, vol. 3, pp. 68–79, March 1960.

[16] R. Bose and D. Ray-Chaudhuri, “Further results on error correcting binary group codes,” *Information and Control*, vol. 3, pp. 279–290, September 1960.

[17] W. Peterson, “Encoding and error correction procedures for the Bose-Chaudhuri codes,” *IEEE Transactions on Information Theory*, vol. IT-6, pp. 459–470, September 1960.

[18] J. Wolf, “Efficient maximum likelihood decoding of linear block codes using a trellis,” *IEEE Transactions on Information Theory*, vol. IT-24, pp. 76–80, January 1978.

[19] B. Honary and G. Markarian, *Trellis Decoding of Block Codes*. Dordrecht, The Netherlands: Kluwer Academic, 1997.

[20] S. Lin, T. Kasami, T. Fujiwara, and M. Fossorier, *Trellises and Trellis-based Decoding Algorithms for Linear Block Codes*. Norwell, MA, USA: Kluwer Academic, 1998.

[21] G. Forney, "Coset Codes-Part II: Binary Lattices and Related Codes," *IEEE Transactions on Information Theory*, vol. 34, pp. 1152–1187, September 1988.

[22] H. Manoukian and B. Honary, "BCJR trellis construction for binary linear block codes," *IEE Proceedings, Communications*, vol. 144, pp. 367–371, December 1997.

[23] B. Honary, G. Markarian, and P. Farrell, "Generalised array codes and their trellis structure," *Electronics Letters*, vol. 29, pp. 541–542, March 1993.

[24] B. Honary and G. Markarian, "Low-complexity trellis decoding of Hamming codes," *Electronics Letters*, vol. 29, pp. 1114–1116, June 1993.

[25] G. M. B. Honary and M. Darnell, "Low-complexity trellis decoding of linear block codes," *IEE Proceedings Communications*, vol. 142, pp. 201–209, August 1995.

[26] Tado Kasami, Toyoo Takata, Toru Fujiwara and Shu Lin, "On Complexity of Trellis Structure of Linear Block Codes," *IEEE Transactions on Information Theory*, vol. 39, pp. 1057–1937, May 1993.

[27] T. Kasami, T. Takata, T. Fujiwara, and S. Lin, "On the optimum bit orders with respect to the state complexity of trellis diagrams for binary linear codes," *IEEE Transactions on Information Theory*, vol. 39, pp. 242–245, January 1993.

[28] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Transactions on Information Theory*, vol. IT-18, pp. 170–182, January 1972.

[29] D. Gorenstein and N. Zierler, "A class of cyclic linear error-correcting codes in p^m synbols," *Journal of the Society of Industrial and Applied Mathematics.*, vol. 9, pp. 107–214, June 1961.

[30] I. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society of Industrial and Applied Mathematics.*, vol. 8, pp. 300–304, June 1960.

[31] E. Berlekamp, "On decoding binary Bose-Chaudhuri-Hocquenghem codes," *IEEE Transactions on Information Theory*, vol. 11, pp. 577–579, 1965.

[32] E. Berlekamp, *Algebraic Coding Theory*. New York, USA: McGraw-Hill, 1968.

[33] J. Massey, "Step-by-step decoding of the Bose-Chaudhuri-Hocquenghem codes," *IEEE Transactions on Information Theory*, vol. 11, pp. 580–585, 1965.

[34] J. Massey, "Shift-register synthesis and BCH decoding," *IEEE Transactions on Information Theory*, vol. IT-15, pp. 122–127, January 1969.

[35] M. Oh and P. Sweeney, "Bit-level soft-decision sequential decoding for Reed Solomon codes," in *Workshop on Coding and Cryptography*, (Paris, France), January 1999.

[36] M. Oh and P. Sweeney, "Low complexity soft-decision sequential decoding using hybrid permutation for RS codes," in *Seventh IMA Conference on Cryptography and Coding*, (Royal Agricultural College, Cirencester, UK), December 1999.

[37] D. Burgess, S. Wesemeyer, and P. Sweeney, "Soft-decision decoding algorithms for RS codes," in *Seventh IMA Conference on Cryptography and Coding*, (Royal Agricultural College, Cirencester, UK), December 1999.

[38] Consultative Committee for Space Data Systems, *Blue Book: Recommendations for Space Data System Standards: Telemetry Channel Coding*, May 1984.

[39] European Telecommunication Standard Institute (ETSI), *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for MVDS at 10GHz and above*, ETS 300 748 ed., October 1996. http://www.etsi.org/.

[40] F. Taylor, "Residue arithmetic: A tutorial with examples," *IEEE Computer Magazine*, vol. 17, pp. 50–62, May 1984.

[41] N. Szabo and R. Tanaka, *Residue Arithmetic and Its Applications to Computer Technology*. New York, USA: McGraw-Hill, 1967.

[42] R. Watson and C. Hastings, "Self-checked computation using residue arithmetic," *Proceedings of the IEEE*, vol. 54, pp. 1920–1931, December 1966.

[43] H. Krishna, K. Y. Lin, and J. D. Sun, "A coding theory approach to error control in redundant residue number systems - part I: Theory and single error correction," *IEEE Transactions on Circuits and Systems*, vol. 39, pp. 8–17, January 1992.

[44] J. D. Sun and H. Krishna, "A coding theory approach to error control in redundant residue number systems — part II: Multiple error detection and correction," *IEEE Transactions on Circuits and Systems*, vol. 39, pp. 18–34, January 1992.

[45] T. H. Liew, L. L. Yang, and L. Hanzo, “Soft-decision Redundant Residue Number System Based Error Correction Coding,” in *Proceedings of IEEE VTC'99*, (Amsterdam, The Netherlands), pp. 2546–2550, September 1999.

[46] G. Ungerboeck, “Treliis-coded modulation with redundant signal sets part 1: Introduction,” *IEEE Communications Magazine*, vol. 25, pp. 5–11, February 1987.

[47] G. Ungerboeck, “Treliis-coded modulation with redundant signal sets part II: State of the art,” *IEEE Communications Magazine*, vol. 25, pp. 12–21, February 1987.

[48] C. Schlegel, *Trellis Coding*. New York, USA: IEEE Press, 1997.

[49] R. Steele and L. Hanzo, eds., *Mobile Radio Communications: Second and Third Generation Cellular and WATM Systems*. New York, USA: IEEE Press - John Wiley & Sons, 2nd ed., 1999.

[50] W. Koch and A. Baier, “Optimum and sub-optimum detection of coded data disturbed by time-varying intersymbol interference,” *IEEE Globecom*, pp. 1679–1684, December 1990.

[51] J. Erfanian, S. Pasupathy, and G. Gulak, “Reduced complexity symbol dectectors with parallel structures for ISI channels,” *IEEE Transactions on Communications*, vol. 42, pp. 1661–1671, 1994.

[52] P. Robertson, E. Villebrun, and P. Höher, “A Comparison of Optimal and Sub-Optimal MAP Decoding Algorithms Operating in the Log Domain,” in *Proceedings of the International Conference on Communications*, (Seattle, USA), pp. 1009–1013, June 1995.

[53] J. Hagenauer and P. Höher, “A Viterbi algorithm with soft-decision outputs and its applications,” in *IEEE Globecom*, pp. 1680–1686, 1989.

[54] J. Hagenauer, “Source-controlled channel decoding,” *IEEE Transactions on Communications*, vol. 43, pp. 2449–2457, September 1995.

[55] S. L. Goff, A. Glavieux, and C. Berrou, “Turbo-codes and high spectral efficiency modulation,” in *Proceedings of IEEE International Conference on Communications*, pp. 645–649, 1994.

[56] U. Wachsmann and J. Huber, “Power and bandwidth efficient digital communications using turbo codes in multilevel codes,” *European Transactions on Telecommunications*, vol. 6, pp. 557–567, September–October 1995.

[57] P. Robertson and T. Wörz, “Bandwidth-Efficient Turbo Trellis-Coded Modulation Using Punctured Component Codes,” *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 206–218, February 1998.

[58] S. Benedetto and G. Montorsi, “Design of parallel concatenated convolutional codes,” *IEEE Transactions on Communications*, vol. 44, pp. 591–600, May 1996.

[59] S. Benedetto and G. Montorsi, “Unveiling turbo codes: Some results on parallel concatenated coding schemes,” *IEEE Transactions on Information Theory*, vol. 42, pp. 409–428, March 1996.

[60] L. Perez, J. Seghers, and D. Costello, “A distance spectrum interpretation of turbo codes,” *IEEE Transactions on Information Theory*, vol. 42, pp. 1698–1709, November 1996.

[61] J. Hagenauer, E. Offer, and L. Papke, “Iterative decoding of binary block and convolutional codes,” *IEEE Transactions on Information Theory*, vol. 42, pp. 429–445, March 1996.

[62] R. M. Pyndiah, “Near-optimum decoding of product codes: Block turbo codes,” *IEEE Transactions on Communications*, vol. 46, pp. 1003–1010, August 1998.

[63] H. Nickl, J. Hagenauer, and F. Burkett, “Approaching Shannon's capacity limit by 0.27 dB using simple Hamming codes,” *IEEE Communications Letters*, vol. 1, pp. 130–132, September 1997.

[64] O. F. Açikel and W. E. Ryan, “Punctured turbo-codes for BPSK/QPSK channels,” *IEEE Transactions on Communications*, vol. 47, pp. 1315–1323, September 1999.

[65] P. Jung and M. Nasshan, “Performance evaluation of turbo codes for short frame transmission systems,” *IEE Electronics Letters*, vol. 30, pp. 111–112, January 1994.

[66] P. Jung, “Comparison of turbo-code decoders applied to short frame transmission systems,” *IEEE Journal on Selected Areas in Communications*, pp. 530–537, 1996.

[67] P. Jung, M. Naßhan, and J. Blanz, “Application of Turbo-Codes to a CDMA Mobile Radio System Using Joint Detection and Antenna Diversity,” in *Proceedings of the IEEE Conference on Vehicular Technology*, pp. 770–774, 1994.

[68] A. Barbulescu and S. Pietrobon, “Interleaver design for turbo codes,” *IEE Electronics Letters*, pp. 2107–2108, December 1994.

[69] B. Sklar, “A Primer on Turbo Code Concepts,” *IEEE Communications Magazine*, pp. 94–102, December 1997.

[70] V. Tarokh, N. Seshadri, and A. R. Calderbank, "Space-Time Codes for High Data Rate Wireless Communication: Performance Criterion and Code Construction," *IEEE Transactions on Information Theory*, vol. 44, pp. 744–765, March 1998.

[71] S. M. Alamouti, "A Simple Transmit Diversity Technique for Wireless Communications," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 1451–1458, October 1998.

[72] V. Tarokh, H. Jafarkhani, and A. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Transactions on Information Theory*, vol. 45, pp. 1456–1467, May 1999.

[73] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, "Space-time block coding for wireless communications: Performance results," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 451–460, March 1999.

[74] G. Bauch, A. Naguib, and N. Seshadri, "MAP Equalization of Space-Time Coded Signals over Frequency Selective Channels," in *Proceedings of Wireless Communications and Networking Conference*, (New Orleans, USA), September 1999.

[75] G. Bauch and N. Al-Dhahir, "Reduced-complexity turbo equalization with multiple transmit and receive antennas over multipath fading channels," in *Proceedings of Information Sciences and Systems*, (Princeton, USA), pp. WP3 13–18, March 2000.

[76] D. Agrawal, V. Tarokh, A. Naguib, and N. Seshadri, "Space-time coded OFDM for high data-rate wireless communication over wideband channels," in *Proceedings of IEEE Vehicular Technology Conference*, (Ottawa, Canada), pp. 2232–2236, May 1998.

[77] Y. Li, N. Seshadri, and S. Ariyavisitakul, "Channel estimation for OFDM systems with transmitter diversity in mobile wireless channels," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 461–471, March 1999.

[78] Y. Li, J. Chuang, and N. Sollenberger, "Transmitter diversity for OFDM systems and its impact on high-rate data wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, pp. 1233–1243, July 1999.

[79] A. Naguib, N. Seshdri, and A. Calderbank, "Increasing Data Rate Over Wireless Channels: Space-Time Coding for High Data Rate Wireless Communications," *IEEE Signal Processing Magazine*, vol. 17, pp. 76–92, May 2000.

[80] A. F. Naguib, V. Tarokh, N. Seshadri, and A. R. Calderbank, "A Space-Time Coding Modem for High-Data-Rate Wireless Communications," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 1459–1478, October 1998.

[81] H. Imai and S. Hirawaki, "A New Multilevel Coding Method Using Error Correcting Codes," *IEEE Transactions on Information Theory*, vol. 23, pp. 371–377, May 1977.

[82] A.R. Calderbank, "Multilevel Codes and Multistage Decoding," *IEEE Transactions on Communications*, vol. 37, pp. 222–229, March 1989.

[83] G. Ungerböck, "Channel Coding with Multilevel/Phase Signals," *IEEE Transactions on Information Theory*, vol. 28, pp. 55–67, January 1982.

[84] E. Zehavi, "8-PSK trellis codes for a Rayleigh fading channel," *IEEE Transactions on Communications*, vol. 40, pp. 873–883, May 1992.

[85] L.-F. Wei, "Trellis-coded modulation with multidimensional constellations," *IEEE Transactions on Information Theory*, vol. 33, pp. 483–501, July 1987.

[86] G. J. Pottie and D. P. Taylor, "Multilevel Codes Based on Partitioning," *IEEE Transactions on Information Theory*, vol. 35, pp. 87–98, January 1989.

[87] N. Seshadri and C-E. W. Sundberg, "Multilevel Trellis Coded Modulations for the Rayleigh Fading Channel," *IEEE Transactions on Communications*, vol. 41, pp. 1300–1310, September 1993.

[88] Y. Kofman, E. Zahavi and S. Shamai, "Performance Analysis of a Multilevel Coded Modulation System," *IEEE Transactions on Communications*, vol. 42, pp. 299–312, February 1994.

[89] J. Huber and U. Wachsmann, "Capacities of Equivalent Channels in Multilevel Coding Schemes," *IEE Electonics Letter*, vol. 30, pp. 557–558, March 1994.

[90] X. Li and J.A. Ritcey, "Bit-interleaved coded modulation with iterative decoding," *IEEE Communications Letters*, vol. 1, November 1997.

[91] G. Caire, G. Taricco, and E. Biglieri, "Bit-Interleaved Coded Modulation," *IEEE Transactions on Information Theory*, vol. 44, pp. 927–946, May 1998.

[92] P. Robertson, T. Wörz, "Bandwidth-Efficient Turbo Trellis-Coded Modulation Using Punctured Component Codes," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 206–218, February 1998.

[93] R. H. Morelos-Zaragoza, M. P. C. Fossorier, S. Lin and H. Imai, "Multilevel Coded Modulation for Unequal Error Protection and Multistage Decoding-Part I: Symmetric Constellations," *IEEE Transactions on Communications*, vol. 48, pp. 204–213, February 2000.

[94] M. Isaka, M. P. C. Fossorier, R. H. Morelos-Zaragoza, S. Lin and H. Imai, "Multilevel Coded Modulation for Unequal Error Protection and Multistage Decoding-Part II: Asymmetric Constellations," *IEEE Transactions on Communications*, vol. 48, pp. 774–786, May 2000.

[95] A. Chindapol and J. A. Ritcey, "Design, Analysis and Performance Evaluation for BICM-ID with Square QAM Constellations in Rayleigh Fading Channels," *IEEE Journal on Selected Areas in Communications*, vol. 19, pp. 944–957, May 2001.

[96] Y. Huang and J. A. Ritcey, "Tight BER Bounds for Iteratively Decoded Bit-Interleaved Space-Time Coded Modulation," *IEEE Communication Letters*, vol. 8, pp. 153–155, March 2004.

[97] J. Hou and M. H. Lee, "Multilevel LDPC Codes Design for Semi-BICM," *IEEE Communications Letters*, pp. 674–676, Nov. 2004.

[98] L. Lampe, R. Schober and R. F. H. Fischer, "Multilevel Coding for Multiple-Antenna Transmission," *IEEE Transactions on Wireless Communications*, vol. 3, pp. 203–208, January 2004.

[99] P. A. Martin, D. M. Rankin and D. P. Taylor, "Multi-Dimensional Space-Time Multilevel Codes," *IEEE Transactions on Wireless Communications*, vol. 5, pp. 3287–3295, November 2006.

[100] A. S. Mohammed, W. Hidayat and M. Bossert, "Multidimensional 16-QAM Constellation Labeling of BI-STCM-ID With the Alamouti Scheme," *IEEE Wireless Communications and Networking Conference (WCNC)*, vol. 3, pp. 1217–1220, April 2006.

[101] A. S. Mohammed, Y. Gong and M. Bossert, "On Multidimensional BICM-ID with 8-PSK Constellation Labeling," *IEEE 18th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pp. 1–5, September 2007.

[102] T. Matsumoto, S. Ibi, S. Sampei and R. Thomä, "Adaptive Transmission With Single-Carrier Multilevel BICM," *Proceedings of the IEEE*, vol. 95, pp. 2354–2367, December 2007.

[103] F. Simoens, H. Wymeersch and M. Moeneclaey, "Linear Precoders for Bit-Interleaved Coded Modulation on AWGN Channels: Analysis and Design Criteria," *IEEE Transactions on Information Theory*, vol. 54, pp. 87–99, January 2008.

[104] R. Battiti and G. Tecchiolli, "Reactive Search : Toward Self-Tuning Heuristics," *Operations Research Society of Americ (ORSA) Journal on Computing*, vol. 6, pp. 126–140, November 1994.

[105] R. Pyndiah, A. Glavieux, A. Picart, and S. Jacq, "Near optimum decoding of product codes," in *GLOBECOM 94*, (San Francisco, USA), pp. 339–343, November 1994.

[106] W. Peterson and E. Weldon Jr., *Error Correcting Codes*. Cambridge, MA, USA: MIT. Press, 2nd ed., August 1972. ISBN: 0262160390.

[107] T. Kasami, *Combinational Mathematics and its Applications*. Darham, NC, USA: University of North Carolina Press, 1969.

[108] W. Peterson, *Error Correcting Codes*. Cambridge, MA, USA: MIT Press, 1st ed., 1961.

[109] F. MacWilliams and J. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam: North-Holland, 1977.

[110] B. Sklar, *Digital Communications—Fundamentals and Applications*. Englewood Cliffs, NJ, USA: Prentice Hall, 1988.

[111] I. Blake, ed., *Algebraic Coding Theory: History and Development*. Dowden, Hutchinson and Ross, 1973.

[112] G. Clark Jr. and J. Cain, *Error Correction Coding for Digital Communications*. New York, USA: Plenum Press, May 1981. ISBN: 0306406152.

[113] V. Pless, *Introduction to the Theory of Error-correcting Codes*. New York, USA: John Wiley & Sons, 1982. ISBN: 0471813044.

[114] R. Blahut, *Theory and Practice of Error Control Codes*. Reading, MA, USA: Addison-Wesley, 1983. ISBN 0-201-10102-5.

[115] C. E. Shannon, *Mathematical Theory of Communication*. University of Illinois Press, 1963.

[116] C. Heegard and S. Wicker, *Turbo Coding*. Kluwer International, 1999.

[117] M. Bossert, *Channel Coding for Telecommunications*. New York, USA: John Wiley & Sons, 1999. ISBN 0-471-98277-6.

[118] B. Vucetic and J. Yuan, *Turbo Codes Principles and Applications*. Dordrecht, The Netherlands: Kluwer Academic, 2000.

[119] R. Lidl and H. Niederreiter, *Finite Fields*. Cambridge, UK: Cambridge University Press, 1996.

[120] S. Lin and D. Constello Jr., *Error Control Coding: Fundamentals and Applications*. Englewood Cliffs, NJ, USA: Prentice Hall, October 1982. ISBN: 013283796X.

[121] A. Michelson and A. Levesque, *Error Control Techniques for Digital Communication*. New York, USA: John Wiley & Sons, 1985.

[122] D. Hoffman, D. Leonard, C. Lindner, K. Phelps, C. Rodger, and J. Wall, *Coding Theory*. New York, USA: Marcel Dekker, 1991.

[123] J. Huber, *Trelliscodierung*. Berlin, Germany: Springer Verlag, 1992.

[124] J. Anderson and S. Mohan, *Source and Channel Coding — An Algorithmic Approach*. Dordrecht, The Netherlands: Kluwer Academic, 1993.

[125] S. Wicker, *Error Control Systems for Digital Communication and Storage*. Englewood Cliffs, NJ, USA: Prentice Hall, 1994.

[126] J. Proakis, *Digital Communications*. New York, USA: McGraw Hill: International Editions, 3rd ed., 1995.

[127] P. Sweeney, *Error Control Coding: An Introduction*. New York, USA: Prentice Hall, 1991.

[128] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design, and iterative decoding," *IEEE Transactions on Information Theory*, vol. 44, pp. 909–926, May 1998.

[129] C. Douillard, A. Picart, M. Jézéquel, P. Didier, C. Berrou, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," *European Transactions on Communications*, vol. 6, pp. 507–511, 1995.

[130] D. Raphaeli and Y. Zarai, "Combined turbo equalization and turbo decoding," *IEEE Communications Letters*, vol. 2, pp. 107–109, April 1998.

[131] R. Y. S. Tee, F. C. Kuo and L. Hanzo, "Multilevel Generalised Low-Density Parity-Check Codes," *IEE Electronics Letters*, vol. 42, pp. 167 – 168, 2 February 2006.

[132] R. Y. S. Tee, S. X. Ng and L. Hanzo, "Three-Dimensional EXIT Chart Analys'is of Iteraltive Detecltion Aided Coded Modulaltion Schemes," in *IEEE Vehicular Technology Conference-Spring (VTC)*, vol. 5, (Melbourne, Austrialia), pp. 2494 – 2498, May 2006.

[133] R. Y. S. Tee, F. C. Kuo and L. Hanzo, "Generalized Low-Density Parity-Check Coding Aided Multilevel Codes," in *IEEE Vehicular Technology Conference-Spring (VTC)*, vol. 5, (Melbourne, Australia), pp. 2398 – 2402, May 2006.

[134] R. Y. S. Tee, T. D. Nguyen, L. L. Yang and L. Hanzo, "Serially Concatenated Luby Transform Coding and Bit-Interleaved Coded Modulation Using Iteratlive Decoding for the Wireless Internet," in *IEEE Vehicular Technology Conference-Spring (VTC)*, vol. 1, (Melbourne, Australia), pp. 22 – 26, May 2006.

[135] R. Y. S. Tee, S. X. Ng and L. Hanzo, "Precoder-aided iterative detection assisted multilevel coding and three-dimensional EXIT-chart analysis," in *IEEE Wireless Communications and Networking Conference (WCNC)*, vol. 3, (Las Vegas, USA), pp. 1322 – 1326, 3-6 April 2006.

[136] R. Y. S. Tee, O. Alamri and L. Hanzo, "Joint Design of Twin-Antenna Assisted Space-Time Multilevel Sphere Packing Aided Coded Modulation," in *IEEE Vehicular Technology Conference-Fall (VTC)*, (Montreal, Canada), pp. 1–5, September 2006.

[137] R. Y. S. Tee, O. Alamri, S. X. Ng and L. Hanzo, "Equivalent-Capacity-Based Design of Space-Time Block-Coded Sphere-Packing-Aided Multilevel Code," in *IEEE Conference on Communications (ICC)*, (Glasgow, UK), pp. 4173 – 4178, 24-28 June 2007.

[138] R. Y. S. Tee, T. D. Nguyen, S. X. Ng, L-L. Yang and L. Hanzo, "Luby Transform Coding Aided Bit-Interleaved Coded Modulation for the Wireless Internet," in *IEEE Vehicular Technology Conference-Fall (VTC)*, (Baltimore, USA), pp. 2025 – 2029, October 2007.

[139] M. El-Hajjar, R. Y. S. Tee, H. Bin, L-L. Yang and L. Hanzo, "Downlink Steered Space-Time Spreading Assisted Generalised Multicarrier DS-CDMA Using Sphere-Packing-Aided Multilevel Coding," in *IEEE Vehicular Technology Conference-Fall (VTC)*, (Baltimore, USA), pp. 472 – 476, October 2007.

[140] R. Y. S. Tee, O. Alamri, S. X. Ng and L. Hanzo, "Bit-Interleaved Sphere-Packing Aided Space-Time Coded Modulation," *IEEE Transactions on Vehicular Technology*, submitted in May 2007.

[141] R. Y. S. Tee, O. Alamri, S. X. Ng and L. Hanzo, “Equivalent Capacity Based Joint Multilevel Coding And Space Time Transmit-Diversity Design,” *IEEE Transactions on Vehicular Technology*, accepted for publication.

[142] R. Y. S. Tee, R. Maunder, J. Wang and L. Hanzo, “Near-Capacity Irregular Bit-Interleaved Coded Modulation,” in *IEEE Vehicular Technology Conference-Spring (VTC)*, (Singapore), May 2008.

[143] R. Y. S. Tee, R. G. Maunder and L. Hanzo, “EXIT-Chart Aided Near-Capacity Irregular Bit-Interleaved Coded Modulation Design,” *IEEE Transactions on Wireless Communications*, submitted in October 2007.

[144] R. Y. S. Tee and L. Hanzo, “Adaptive Iteratively Detected Multilevel Generalised Low-Density Parity-Check Codes,” in *IEEE Vehicular Technology Conference-Spring (VTC)*, (Calgary, Canada), submitted in February 2008.

[145] J. Heller and I. Jacobs, “Viterbi decoding for satellite and space communication,” *IEEE Transactions on Communication Technology*, vol. COM-19, pp. 835–848, October 1971.

[146] C. Berrou, P. Adde, E. Angui, and S. Faudeil, “A low complexity soft-output Viterbi decoder architecture,” in *Proceedings of the International Conference on Communications*, pp. 737–740, May 1993.

[147] R. Pyndiah, “Iterative decoding of product codes: Block turbo codes,” in *International Symposium on Turbo Codes and related topics*, (Brest, France), pp. 71–79, September 1997.

[148] M. Breiling and L. Hanzo, “Optimum non-iterative decoding of turbo codes,” *IEEE Transactions on Information Theory*, vol. 46, pp. 2212–2228, September 2000.

[149] M. Breiling and L. Hanzo, “Optimum Non-iterative Turbo-Decoding,” *Proceedings of PIMRC’97, Helsinki, Finland*, pp. 714–718, September 1997.

[150] P. Robertson, “Illuminating the structure of code and decoder of parallel concatenated recursive systematic (turbo) codes,” *IEEE Globecom*, pp. 1298–1303, 1994.

[151] M. Breiling, “Turbo coding simulation results,” tech. rep., Universit$\ddot{a}$t Karlsruhe, Germany and Southampton University, UK, 1997.

[152] C. Berrou, “Some clinical aspects of turbo codes,” in *International Symposium on Turbo Codes and related topics*, (Brest, France), pp. 26–31, September 1997.

[153] J. Erfanian, S. Pasupathy, and G. Gulak, “Low-complexity symbol detectors for isi channels,” in *IEEE Globecom*, (San Diego), December 1990.

[154] A. Viterbi, “Approaching the Shannon limit: Theorist’s dream and practitioner’s challenge,” in *Proceedings of the International Conference on Millimeter Wave and Far Infrared Science and Technology*, pp. 1–11, 1996.

[155] A. J. Viterbi, “An Intuitive Justification and a Simplified Implementation of the MAP Decoder for Convolutional Codes,” *IEEE Journal on Selected Areas in Communications*, pp. 260–264, February 1997.

[156] J. Woodard, T. Keller, and L. Hanzo, “Turbo-coded orthogonal frequency division multiplex transmission of 8 kbps encoded speech,” in *Proceedings of ACTS Mobile Communication Summit ’97*, (Aalborg, Denmark), pp. 894–899, ACTS, 7–10 October 1997.

[157] P. Jung and M. Nasshan, “Dependence of the error performance of turbo-codes on the interleaver structure in short frame transmission systems,” *IEE Electronics Letters*, pp. 287–288, February 1994.

[158] P. Jung and M. Naßhan, “Results on Turbo-Codes for Speech Transmission in a Joint Detection CDMA Mobile Radio System with Coherent Receiver Antenna Diversity,” *IEEE Transactions on Vehicular Technology*, vol. 46, pp. 862–870, November 1997.

[159] H. Herzberg, “Multilevel Turbo Coding with Short Interleavers,” *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 303–309, February 1998.

[160] T.A. Summmers and S.G. Wilson, “SNR Mismatch and Online Estimation in Turbo Decoding,” *IEEE Transactions on Communications*, vol. 46, pp. 421–423, April 1998.

[161] J. Cavers, “An analysis of pilot symbol assisted modulation for Rayleigh fading channels,” *IEEE Transactions on Vehicular Technology*, vol. 40, pp. 686–693, November 1991.

[162] P. Adde, R. Pyndiah, O. Raoul, and J. R. Inisan, “Block turbo decoder design,” in *International Symposium on Turbo Codes and related topics*, (Brest, France), pp. 166–169, September 1997.

[163] A. Goalic and R. Pyndiah, “Real-time turbo-decoding of product codes on a digital signal processor,” in *International Symposium on Turbo Codes and related topics*, (Brest, France), pp. 267–270, September 1997.

[164] R. Steele and L. Hanzo, eds., *Mobile Radio Communications*, ch. 4.4.4, pp. 425–428. Piscataway, NJ, USA: IEEE Press and Pentech Press, 1999.

[165] F. J. Macwilliams and N. J. A. Sloane, *The theory of error correcting codes*, vol. 16, ch. 18, pp. 567–580. Bell Laboratories, Murray Hill, NJ 07974 USA: North-Holland publishing company, 1978.

[166] S. Ng, T. Liew, L. Yang, and L. Hanzo, “Turbo coding performance using block component codes,” in *Proceedings of VTC 2000 Spring*, (Tokyo, Japan), pp. 849–853, 15–18 May 2000.

[167] J. Blogh and L. Hanzo, *3G Systems and Intelligent Networking*. New York, USA: John Wiley and IEEE Press, 2002. (For detailed contents, please refer to http://www-mobile.ecs.soton.ac.uk.).

[168] L. Hanzo and B. Y. T.H. Liew, *Turbo Coding, Turbo Equalisation and Space-Time Coding*. New York, USA: John Wiley, IEEE Press, 2002. (For detailed contents, please refer to http://www-mobile.ecs.soton.ac.uk.).

[169] G. J. Foschini, Jr., “Layered Space-time architecture for wireless communication in a fading environment when using multi-element antennas,” *Bell Labs Technical Journal*, pp. 41–59, 1996.

[170] J. H. Winters, “Smart antennas for wireless systems,” *IEEE Personal Communications*, vol. 5, pp. 23–27, February 1998.

[171] R. Derryberry, S. Gray, D. Ionescu, G. Mandyam, and B. Raghothaman, “Transmit diversity in 3g cdma systems,” *IEEE Communications Magazine*, vol. 40, pp. 68–75, April 2002.

[172] A. Molisch, M. Win, and J. Winters, “Space-time-frequency (stf) coding for mimo-ofdm systems,” *IEEE Communications Letters*, vol. 6, pp. 370–372, September 2002.

[173] A. Molisch, M. Steinbauer, M. Toeltsch, E. Bonek, and R. Thoma, “Capacity of mimo systems based on measured wireless channels,” *IEEE Journal on Selected Areas in Communications*, vol. 20, pp. 561–569, April 2002.

[174] D. Gesbert, M. Shafi, D.-S. Shiu, P. Smith, and A. Naguib, “From theory to practice: an overview of mimo space-time coded wireless systems,” *IEEE Journal on Selected Areas in Communications*, vol. 21, pp. 281–302, April 2003.

[175] M. Shafi, D. Gesbert, D.-S. Shiu, P. Smith, and W. Tranter, “Guest editorial: Mimo systems and applications,” *IEEE Journal on Selected Areas in Communications*, vol. 21, pp. 277–280, April 2003.

[176] P. Chaudhury, W. Mohr, and S. Onoe, “The 3GPP proposal for IMT-2000,” *IEEE Communications Magazine*, vol. 37, pp. 72–81, December 1999.

[177] G. Foschini Jr. and M. Gans, “On limits of wireless communication in a fading environment when using multiple antennas,” *Wireless Personal Communications*, vol. 6, pp. 311–335, March 1998.

[178] B. Glance and L. Greestein, “Frequency-selective fading effects in digital mobile radio with diversity combining,” *IEEE Transactions on Communications*, vol. COM-31, pp. 1085–1094, September 1983.

[179] F. Adachi and K. Ohno, “BER performance of QDPSK with postdetection diversity reception in mobile radio channels,” *IEEE Transactions on Vehicular Technology*, vol. 40, pp. 237–249, February 1991.

[180] H. Zhou, R. Deng, and T. Tjhung, “Performance of combined diversity reception and convolutional coding for QDPSK land mobile radio,” *IEEE Transactions on Vehicular Technology*, vol. 43, pp. 499–508, August 1994.

[181] J. Winters, “Switched diversity with feedback for DPSK mobile radio systems,” *IEEE Transactions on Information Theory*, vol. 32, pp. 134–150, February 1983.

[182] G. Raleigh and J. Cioffi, “Spatio-temporal coding for wireless communications,” in *GLOBECOM ’96*, (London, UK), pp. 533–537, November 1996.

[183] A. Wittneben, “Base station modulation diversity for digital SIMULCAST,” in *Proceedings of IEEE Vehicular Technology Conference*, pp. 505–511, May 1993.

[184] N. Seshadri and J. Winters, “Two signalling schemes for improving the error performance of frequency-division-duplex (FDD) transmission systems using transmitter antenna diversity,” *International Journal of Wireless Information Networks*, vol. 1, pp. 49–60, January 1994.

[185] J. Winters, “The diversity gain of transmit diversity in wireless systems with Rayleigh fading,” *IEEE Transactions on Vehicular Technology*, vol. 47, pp. 119–123, February 1998.

[186] T. Hattori and K. Hirade, “Multitransmitter simulcast digital signal transmission by using frequency offset strategy in land mobile radio-telephone system,” *IEEE Transactions on Vehicular Technology*, vol. 27, pp. 231–238, 1978.

[187] A. Hiroike, F. Adachi, and N. Nakajima, “Combined effects of phase sweeping transmitter diversity and channel coding,” *IEEE Transactions on Vehicular Technology*, vol. 41, pp. 170–176, May 1992.

[188] N. Seshadri, V. Tarokh, and A. Calderbank, “Space-Time Codes for High Data Rate Wireless Communications: Code Construction,” in *Proceedings of IEEE Vehicular Technology Conference ’97*, (Phoenix, Arizona), pp. 637–641, 1997.

[189] V. Tarokh and N. Seshadri and A. Calderbank, “Space-time codes for high data rate wireless communications: Performance criterion and code construction,” in *Proceedings of the IEEE International Conference on Communications ’97*, (Montreal, Canada), pp. 299–303, 1997.

[190] V. Tarokh, A. Naguib, N. Seshadri, and A. Calderbank, “Space-time codes for high data rate wireless communications: Mismatch analysis,” in *Proceedings of the IEEE International Conference on Communications ’97*, (Montreal, Canada), pp. 309–313, 1997.

[191] V. Tarokh, A. Naguib, N. Seshadri, and A. R. Calderbank, “Space-time codes for high data rate wireless communication: Performance criteria in the presence of channel estimation errors, mobility, and multile paths,” *IEEE Transactions on Communications*, vol. 47, pp. 199–207, February 1999.

[192] D. Brennan, “Linear diversity combining techniques,” *Proceedings of the IRE*, vol. 47, pp. 1075–1102, 1959.

[193] L. Hanzo, W. Webb, and T. Keller, *Single- and Multi-Carrier Quadrature Amplitude Modulation: Principles and Applications for Personal Communications, WLANs and Broadcasting*. IEEE Press, 2000.

[194] G. Bauch, “Concatenation of space-time block codes and Turbo-TCM,” in *Proceedings of IEEE International Conference on Communications*, (Vancouver, Canada), pp. 1202–1206, June 1999.

[195] G. Forney, “Convolutional codes: I. Algebraic structure,” *IEEE Transactions on Information Theory*, vol. 16, pp. 720–738, November 1970.

[196] G. Forney, “Burst-correcting codes for the classic burst channel,” *IEEE Transactions on Communication Technology*, vol. COM-19, pp. 772–781, October 1971.

[197] W. Jakes Jr., ed., *Microwave Mobile Communications*. New York, USA: John Wiley & Sons, 1974.

[198] J. Hagenauer, “Rate-compatible puncture convolutional codes (RCPC) and their application,” *IEEE Transactions on Communications*, vol. 36, pp. 389–400, April 1988.

[199] S. Red, M. Oliphant, and M. Weber, *An Introduction to GSM*. London, UK: Artech House, 1995.

[200] 3GPP, *Multiplexing and channel coding (TDD)*. 3G TS 25.222, http://www.3gpp.org.

[201] T. Ojanperä and R. Prasad, *Wideband CDMA for Third Generation Mobile Communications*. London, UK: Artech House, 1998.

[202] S. Al-Semari and T. Fuja, “I-Q TCM: Reliable communication over the Rayleigh fading channel close to the cutoff rate,” *IEEE Transactions on Information Theory*, vol. 43, pp. 250–262, January 1997.

[203] R. Horn and C. Johnson, *Matrix Analysis*. New York, USA: Cambridge University Press, 1988.

[204] W. Choi and J. Cioffi, “Space-Time Block Codes over Frequency Selective Fading Channels,” in *Proceedings of VTC 1999 Fall*, (Amsterdam, Holland), pp. 2541–2545, 19–22 September 1999.

[205] Z. Liu, G. Giannakis, A. Scaglione, and S. Barbarossa, “Block precoding and transmit-antenna diversity for decoding and equalization of unknown multipath channels,” in *Proceedings of the 33rd Asilomar Conference Signals, Systems and Computers*, (Pacific Grove, Canada), pp. 1557–1561, 1–4 November 1999.

[206] Z. Liu and G. Giannakis, “Space-time coding with transmit antennas for multiple access regardless of frequency-selective multipath,” in *Proceedings of the 1st Sensor Array and Multichannel SP Workshop*, (Boston, USA), 15–17 March 2000.

[207] T. Liew, J. Pliquett, B. Yeap, L. L. Yang, and L. Hanzo, “Comparative study of space time block codes and various concatenated turbo coding schemes,” in *PIMRC 2000*, (London, UK), pp. 741–745, 18–21 September 2000.

[208] T. Liew, J. Pliquett, B. Yeap, L. L. Yang, and L. Hanzo, “Concatenated space time block codes and TCM, turbo TCM, convolutional as well as turbo codes,” in *GLOBECOM 2000*, (San Francisco, USA), 27 November – 1 December 2000.

[209] R. Steele and W. Webb, “Variable rate QAM for data transmission over Rayleigh fading channels,” in *Proceeedings of Wireless ’91*, (Calgary, Alberta), pp. 1–14, IEEE, 1991.

[210] W. Webb and R. Steele, “Variable rate QAM for mobile radio,” *IEEE Transactions on Communications*, vol. 43, pp. 2223–2230, July 1995.

[211] J. M. Torrance and L. Hanzo, “Latency and Networking Aspects of Adaptive Modems over Slow Indoors Rayleigh Fading Channels,” *IEEE Transactions on Vehicular Technology*, vol. 48, no. 4, pp. 1237–1251, 1998.

[212] J. Torrance and L. Hanzo, “Performance upper bound of adaptive QAM in slow Rayleigh-fading environments,” in *Proceedings of IEEE ICCS’96/ISPACS’96*, (Singapore), pp. 1653–1657, IEEE, 25–29 November 1996.

[213] J. M. Torrance, L. Hanzo, and T. Keller, “Interference aspects of adaptive modems over slow Rayleigh fading channels,” *IEEE Transactions on Vehicular Technology*, vol. 48, pp. 1527–1545, September 1999.

[214] T. Keller and L. Hanzo, “Adaptive orthogonal frequency division multiplexing schemes,” in *Proceeding of ACTS Mobile Communication Summit ’98*, (Rhodes, Greece), pp. 794–799, ACTS, 8–11 June 1998.

[215] T. Keller and L. Hanzo, “Blind-detection assisted sub-band adaptive turbo-coded OFDM schemes,” in *Proceedings of VTC’99 (Spring)*, (Houston, USA), pp. 489–493, IEEE, 16–20 May 1999.

[216] H. Matsuoka, S. Sampei, N. Morinaga, and Y. Kamio, "Adaptive modulation systems with variable coding rate concatenated code for high quality multi-media communication systems," in *Proceedings of IEEE VTC'96*, (Atlanta, USA), pp. 487–491, 28 April–1 May 1996.

[217] S. G. Chua and A. J. Goldsmith, "Variable-rate variable-power mQAM for fading channels," in *Proceedings of IEEE VTC'96*, (Atlanta, USA), pp. 815–819, 28 April–1 May 1996.

[218] V. Lau and M. Macleod, "Variable rate adaptive trellis coded QAM for high bandwidth efficiency applications in Rayleigh fading channels," in *Proceedings of IEEE Vehicular Technology Conference (VTC'98)*, (Ottawa, Canada), pp. 348–352, IEEE, 18–21 May 1998.

[219] I. Kalet, "The multitone channel," *IEEE Transactions on Communications*, vol. 37, pp. 119–124, February 1989.

[220] J. Torrance and L. Hanzo, "Optimisation of switching levels for adaptive modulation in a slow Rayleigh fading channel," *Electronics Letters*, vol. 32, pp. 1167–1169, 20 June 1996.

[221] J. Torrance and L. Hanzo, "On the upper bound performance of adaptive QAM in a slow Rayleigh fading," *IEE Electronics Letters*, pp. 169–171, April 1996.

[222] L. Hanzo, C. H. Wong, and M. S. Yee, *Adaptive Wireless Transceivers*. New York, USA: John Wiley, IEEE Press, 2002. (For detailed contents, please refer to http://www-mobile.ecs.soton.ac.uk.).

[223] C. Wong and L. Hanzo, "Upper-bound performance of a wideband burst-by-burst adaptive modem," *IEEE Transactions on Communications*, vol. 48, pp. 367–369, March 2000.

[224] S. Sampei, S. Komaki, and N. Morinaga, "Adaptive Modulation/TDMA Scheme for Large Capacity Personal Multi-Media Communication Systems," *IEICE Transactions on Communications (Japan)*, vol. E77-B, pp. 1096–1103, September 1994.

[225] T. Keller, T. H. Liew, and L. Hanzo, "Adaptive redundant residue number system coded multicarrier modulation," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 2292–2301, November 2000.

[226] M. Yee, T. Liew, and L. Hanzo, "Burst-by-burst adaptive turbo coded radial basis function assisted feedback equalisation," *IEEE Transactions on Communications*, vol. 49, November 2001.

[227] V. Lau and M. Macleod, "Variable-rate adaptive trellis coded QAM for flat-fading channels," *IEEE Transactions on Communications*, vol. 49, pp. 1550–1560, September 2001.

[228] V. Lau and S. Maric, "Variable rate adaptive modulation for DS-CDMA," *IEEE Transactions on Communications*, vol. 47, pp. 577–589, April 1999.

[229] S. Chua and A. Goldsmith, "Adaptive Coded Modulation for Fading Channels," *IEEE Transactions on Communications*, vol. 46, pp. 595–602, May 1998.

[230] X. Liu, P. Ormeci, R. Wesel, and D. Goeckel, "Bandwidth-efficient, low-latency adaptive coded modulation schemes for time-varying channels," in *Proceedings of IEEE International Conference on Communications*, (Helsinki, Finland), June 2001.

[231] C. Tidestav, A. Ahlén and M. Sternad, "Realizable MIMO Decision Feedback Equalizer," *in Proceedings of the 1999 International Conference on Acoustics Speech and Signal Processing*, pp. 2591–2594, 1999.

[232] C. Tidestav, A. Ahlén and M. Sternad, "Realiazable MIMO Decision Feedback Equalizer: Structure and Design," *IEEE Transactions on Signal Processing*, vol. 49, pp. 121–133, January 2001.

[233] C. Tidestav, M. Sternad, and A. Ahlén, "Reuse within a cell - interference rejection or multiuser detection," *IEEE Transactions on Communications*, vol. 47, pp. 1511–1522, October 1999.

[234] T. Liew and L. Hanzo, "Space-time block coded adaptive modulation aided OFDM," in *IEEE Globecom 2001*, (San Antonio, USA), pp. 136–140, 25–29 November 2001.

[235] J. Cheung and R. Steele, "Soft-decision feedback equalizer for continuous-phase modulated signals in wide-band mobile radio channels," *IEEE Transactions on Communications*, vol. 42, pp. 1628–1638, February/March/April 1994.

[236] T. Liew and L. Hanzo, "Space-time codes and concatenated channel codes for wireless communications," *Proceedings of the IEEE*, vol. 90, pp. 183–219, February 2002.

[237] "COST207: Digital land mobile radio communications, final report," tech. rep., Luxembourg, 1989.

[238] J. Torrance and L. Hanzo, "Demodulation level selection in adaptive modulation," *Electronics Letters*, vol. 32, pp. 1751–1752, 12 September 1996.

[239] A. Klein, R. Pirhonen, J. Skoeld, and R. Suoranta, "FRAMES Multiple Access MODE 1 — Wideband TDMA with and without Spreading," in *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, vol. 1, (Helsinki, Finland), pp. 37–41, 1–4 September 1997.

[240] B. J. Choi, T. H. Liew, and L. Hanzo, "Concatenated space-time block coded and turbo coded symbol-by-symbol adaptive OFDM and multi-carrier CDMA systems," in *Proceedings of IEEE VTC 2001-Spring*, p. 528, IEEE, May 2001.

[241] J. Anderson, T. Aulin, and C. Sundberg, *Digital phase modulation*. New York, USA: Plenum Press, 1986.

[242] K. Murota and K. Hirade, "GMSK modulation for digital mobile radio telephony," *IEEE Transactions on Communications*, vol. 29, pp. 1044–1050, July 1981.

[243] ETSI, *Digital Cellular Telecommunications System (Phase 2+); High Speed Circuit Switched Data (HSCSD) — Stage 1; (GSM 02.34 Version 5.2.1)*. European Telecommunications Standards Institute, Sophia Antipolis, Cedex, France, July 1997.

[244] ETSI, *Digital Cellular Telecommunications System (Phase 2+); General Packet Radio Service (GPRS); Overall Description of the GPRS Radio Interface, Stage 2 (GSM 03.64 Version 5.2.0)*. European Telecommunications Standards Institute, Sophia Antipolis, Cedex, France, January 1998.

[245] I. Gerson, M. Jasiuk, J. M. Muller, J. Nowack, and E. Winter, "Speech and channel coding for the half-rate GSM channel," *Proceedings ITG-Fachbericht*, vol. 130, pp. 225–233, November 1994.

[246] R. Salami, C. Laflamme, B. Besette, J. P. Adoul, K. Jarvinen, J. Vainio, P. Kapanen, T. Hankanen, and P. Haavisto, "Description of the GSM enhanced full rate speech codec," in *Proceedings of ICC'97*, 1997.

[247] G. Bauch and V. Franz, "Iterative equalisation and decoding for the GSM-system," in *Proceedings of IEEE Vehicular Technology Conference (VTC'98)*, (Ottawa, Canada), pp. 2262–2266, IEEE, 18–21 May 1998.

[248] F. Burkert, G. Caire, J. Hagenauer, T. Hidelang, and G. Lechner, "Turbo decoding with unequal error protection applied to GSM speech coding," in *Proceedings of IEEE Global Telecommunications Conference, Globecom 96*, (London, UK), pp. 2044–2048, IEEE, 18–22 November 1996.

[249] D. Parsons, *The Mobile Radio Propagation Channel*. London: Pentech Press, 1992.

[250] S. Saunders, *Antennas and Propagation for Wireless Communication Systems Concept and Design*. New York, USA: John Wiley & Sons, 1999.

[251] T. Rappaport, *Wireless Communications Principles and Practice*. Englewood Cliffs, NJ, USA: Prentice Hall, 1996.

[252] ETSI, *GSM Recommendation 05.05, Annex 3*, November 1988.

[253] A. Carlson, *Communication Systems*. New York, USA: McGraw-Hill, 1975.

[254] T. Aulin and C.-E. Sundberg, "CPM-An efficient constant amplitude modulation scheme," *International Journal of Satellite Communications*, vol. 2, pp. 161–186, 1994.

[255] R. Debuda, "Coherent demodulation of frequency shift keying with low deviation ratio," *IEEE Transactions on Communications*, vol. COM-20, pp. 429–435, June 1972.

[256] S. Pasupathy, "Minimum shift keying: A spectrally efficient modulation," *IEEE Communications Magazine*, vol. 17, pp. 14–22, July 1979.

[257] M. K. Simon and C. Wang, "Differential detection of Gaussian MSK in a mobile radio environment," *IEEE Transactions on Vehicular Technology*, vol. 33, pp. 307–320, November 1984.

[258] E. Kreyszig, *Advanced engineering mathematics*. New York, USA: John Wiley & Sons, 7th ed., 1993.

[259] J. Proakis and D. Manolakis, *Digital Signal Processing — Principles, Algorithms and Applications*. Macmillan, 1992.

[260] M. Moher, "Decoding via cross-entropy minimization," in *Proceedings of the IEEE Global Telecommunications Conference 1993*, (Houston, USA), pp. 809–813, 29 November–2 December 1993.

[261] D. A. Johnson, S. W. Wales, and P. H. Waters, "Equalisers for GSM," *IEE Colloquium (Digest)*, no. 21, pp. 1/1–1/6, 1990.

[262] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*. New York USA: McGraw-Hill, 3 ed., 1991.

[263] P. Robertson and T. Wörz, "Coded modulation scheme employing turbo codes," *IEE Electronics Letters*, vol. 31, pp. 1546–1547, 31 August 1995.

[264] "COST207: Digital land mobile radio communications, final report." Office for Official Publications of the European Communities, 1989. Luxembourg.

[265] "GSM Recommendation 05.03: Channel coding," November 1988.

[266] M. Mouly and M. Pautet, *The GSM System for Mobile Communications*. Michel Mouly and Marie-Bernadette Pautet, 1992.

[267] G. Bauch, H. Khorram, and J. Hagenauer, "Iterative equalization and decoding in mobile communications systems," in *European Personal Mobile Communications Conference*, (Bonn, Germany), pp. 301–312, 30 September–2 October 1997.

[268] M. Gertsman and J. Lodge, "Symbol-by-symbol MAP demodulation of CPM and PSK signals on Rayleigh flat-fading channels," *IEEE Transactions on Communications*, vol. 45, pp. 788–799, July 1997.

[269] I. Marsland, P. Mathiopoulos, and S. Kallel, "Non-coherent turbo equalization for frequency selective Rayleigh fast fading channels," in *International Symposium on Turbo Codes and related topics*, (Brest, France), pp. 196–199, September 1997.

[270] Q. Dai and E. Shwedyk, "Detection of bandlimited signals over frequency selective Rayleigh fading channels," *IEEE Transactions on Communications*, pp. 941–950, February/March/April 1994.

[271] F. Jordan and K. Kammeyer, "Study on iterative decoding techniques applied to GSM full-rate channels," in *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*, (Pisa, Italy), pp. 1066–1070, 29 September–2 October 1998.

[272] G. Qin, S. Zhou, and Y. Yao, "Iterative decoding of GMSK modulated convolutional code," *IEE Electronics Letters*, vol. 35, pp. 810–811, 13 May 1999.

[273] K. Narayanan and G. Stüber, "A serial concatenation approach to iterative demodulation and decoding," *IEEE Transactions on Communications*, vol. 47, pp. 956–961, July 1999.

[274] L. Lin and R. Cheng, "Improvements in SOVA-based decoding for turbo codes," in *Proceedings of the IEEE International Conference on Communications*, vol. 3, (Montreal, Canada), pp. 1473–1478, 8–12 June 1997.

[275] Y. Liu, M. Fossorier, and S. Lin, "MAP algorithms for decoding linear block code based on sectionalized trellis diagrams," in *Proceedings of the IEEE Global Telecommunications Conference 1998*, vol. 1, (Sydney, Australia), pp. 562–566, 8–12 November 1998.

[276] Y. V. Svirid and S. Riedel, "Threshold decoding of turbo-codes," *Proceedings of the IEEE International Symposium on Information Theory*, p. 39, September 1995.

[277] J. S. Reeve, "A parallel Viterbi decoding algorithm," *Concurrency and Computation: Practice and Experience*, vol. 13, pp. 95–102, July 2001.

[278] ETSI, *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for 11/12 GHz Satellite Services*, August 1997. ETS 300 421.

[279] A. Knickenberg, B. Yeap, J. Hàmorskỳ, M. Breiling, and L. Hanzo, "Non-iterative Joint Channel Equalisation and Channel Decoding," *IEE Electronics Letters*, vol. 35, pp. 1628–1630, 16 September 1999.

[280] M. S. Yee, B. L. Yeap, and L. Hanzo, "Radial basis function assisted turbo equalisation," in *Proceedings of IEEE Vehicular Technology Conference*, (Japan, Tokyo), pp. 640–644, IEEE, 15–18 May 2000.

[281] A. Glavieux, C. Laot, and J. Labat, "Turbo equalization over a frequency selective channel," in *Proceedings of the International Symposium on Turbo Codes*, (Brest, France), pp. 96–102, 3–5 September 1997.

[282] C. Wong, B. Yeap, and L. Hanzo, "Wideband Burst-by-Burst Adaptive Modulation with Turbo Equalization and Iterative Channel Estimation," in *Proceedings of the IEEE Vehicular Technology Conference 2000*, 2000.

[283] M. Sandell, C. Luschi, P. Strauch, and R. Yan, "Iterative channel estimation using soft decision feedback," in *Proceedings of the Global Telecommunications Conference 1998*, (Sydney, Australia), pp. 3728–3733, 8–12 November 1998.

[284] S. Haykin, *Digital Communications*. New York, USA: John Wiley & Sons, 1988.

[285] S. Benedetto, D. Divsalar, G. Motorsi, and F. Pollara, "A soft-input soft-output APP module for iterative decoding of concatenated codes," *IEEE Communication Letters*, pp. 22–24, 1997.

[286] P. Robertson, P. Höher, and E. Villebrun, "Optimal and sub-optimal maximum a posteriori algorithms suitable for turbo decoding," *European Transactions on Telecommunications*, vol. 8, pp. 119–125, March/April 1997.

[287] A. Whalen, *Detection of signals in noise*. New York, USA: Academic Press, 1971.

[288] C. H. Wong, *Wideband Adaptive Full Response Multilevel Transceivers and Equalizers*. PhD thesis, University of Southampton, UK, November 1999.

[289] B. L. Yeap, C. H. Wong, and L. Hanzo, "Reduced complexity in-phase/quadrature-phase turbo equalisation with iterative channel estimation," in *IEEE International Communications Conference 2001*, (Helsinki, Finland), 11–15 June 2001. Accepted for publication.

[290] G. Ungerböck, “Channel Coding with Multilevel/Phase Signals,” *IEEE Transactions on Information Theory*, vol. IT-28, pp. 55–67, January 1982.

[291] D. Divsalar and M. K. Simon, “The design of trellis coded MPSK for fading channel: Performance criteria,” *IEEE Transactions on Communications*, vol. 36, pp. 1004–1012, September 1988.

[292] D. Divsalar and M. K. Simon, “The design of trellis coded MPSK for fading channel: Set partitioning for optimum code design,” *IEEE Transactions on Communications*, vol. 36, pp. 1013–1021, September 1988.

[293] X. Li and J.A. Ritcey, “Trellis-Coded Modulation with Bit Interleaving and Iterative Decoding,” *IEEE Journal on Selected Areas in Communications*, vol. 17, April 1999.

[294] X. Li and J.A. Ritcey, “Bit-interleaved coded modulation with iterative decoding — Approaching turbo-TCM performance without code concatenation,” in *Proceedings of CISS 1998*, (Princeton University, USA), March 1998.

[295] S. X. Ng, T. H. Liew, L. L. Yang, and L. Hanzo, “Comparative Study of TCM, TTCM, BICM and BICM-ID schemes,” *IEEE Vehicular Technology Conference*, p. 265 (CDROM), May 2001.

[296] S. X. Ng, C. H. Wong and L. Hanzo, “Burst-by-Burst Adaptive Decision Feedback Equalized TCM, TTCM, BICM and BICM-ID,” *International Conference on Communications (ICC)*, pp. 3031–3035, June 2001.

[297] C. S. Lee, S. X. Ng, L. Piazzo and L. Hanzo, “TCM, TTCM, BICM and Iterative BICM Assisted OFDM-Based Digital Video Broadcasting to Mobile Receivers,” *IEEE Vehicular Technology Conference*, p. 113 (CDROM), May 2001.

[298] X. Li and J.A. Ritcey, “Bit-interleaved coded modulation with iterative decoding using soft feedback,” *IEE Electronics Letters*, vol. 34, pp. 942–943, May 1998.

[299] G. Ungerböck, “Trellis-coded modulation with redundant signal sets. Part 1 and 2,” *IEEE Communications Magazine*, vol. 25, pp. 5–21, February 1987.

[300] J. H. Chen and A. Gersho, “Gain-adaptive vector quantization with application to speech coding,” *IEEE Transactions on Communications*, vol. 35, pp. 918–930, September 1987.

[301] R. Blahut, *Theory and Practice of Error Control Codes*, ch. 6, pp. 130–160. IBM Corporation, Owego, NY 13827, USA: Addison-Wesley, 1983.

[302] S. S. Pietrobon, G. Ungerböck, L. C. Perez and D. J. Costello, “Rotationally invariant nonlinear trellis codes for two-dimensional modulation,” *IEEE Transactions on Information Theory*, vol. IT-40, pp. 1773–1791, November 1994.

[303] C. Schlegel, “Chapter 3: Trellis Coded Modulation,” in *Trellis Coding*, pp. 43–89, New York, USA: IEEE Press, September 1997.

[304] J. K. Cavers and P. Ho, “Analysis of the Error Performance of Trellis-Coded Modulations in Rayleigh-Fading Channels,” *IEEE Transactions on Communications*, vol. 40, pp. 74–83, January 1992.

[305] G. D. Forney, “The Viterbi ALgorithm,” *Proceedings of the IEEE*, vol. 61, pp. 268–277, March 1973.

[306] J.G. Proakis, “Optimum Receivers for the Additive White Gaussian Noise Channel,” in *Digital Communications*, pp. 260–274, New York, USA: McGraw-Hill International Edition, 3rd Ed., September 1995.

[307] K. Abend and B. D. Fritchman, “Statistical detection for communication channels with intersymbol interference,” *Proceedings of the IEEE*, vol. 58, pp. 779–785, May 1970.

[308] L. Piazzo, “An Algorithm for SBS Receivers/Decoders,” *IEE Electronics Letters*, vol. 32, pp. 1058–1060, June 1996.

[309] S. S. Pietrobon, R. H. Deng, A. Lafanechére, G. Ungerböck, and D. J. Costello, “Trellis-Coded Multidimensional Phase Modulation,” *IEEE Transactions on Information Theory*, vol. 36, pp. 63–89, January 1990.

[310] L. F. Wei, “Trellis-coded modulation with multidimensional constellations,” *IEEE Transactions on Information Theory*, vol. IT-33, pp. 483–501, July 1987.

[311] P. Robertson, “An Overview of Bandwidth Efficient Turbo Coding Schemes,” in *International Symposium on Turbo Codes and related topics*, (Brest, France), pp. 103–110, September 1997.

[312] L. Lee, “New rate-compatible puncture convolutional codes for Viterbi decoding,” *IEEE Transactions on Communications*, vol. 42, pp. 3073–3079, December 1994.

[313] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “A Soft-Input Soft-Output APP Module for Iterative Decoding of Concatenated Codes,” *IEEE Communications Letters*, vol. 1, pp. 22–24, January 1997.

[314] J.G. Proakis, “Chapter 10: Communication Through Band-Limited Channels,” in *Digital Communications*, pp. 583–635, New York, USA: McGraw-Hill International Editions, 3rd Ed., September 1995.

[315] D. F. Mix, *Random Signal Processing*. Englewood Cliffs, NJ, USA: Prentice Hall, 1995.

[316] J. C. Cheung, *Adaptive Equalisers for Wideband TDMA Mobile Radio*. PhD thesis, Department of Electronics and Computer Science, University of Southampton, UK, 1991.

[317] J. M. Torrance and L. Hanzo, "Interference aspects of adaptive modems over slow Rayleigh fading channels," *IEEE Vehicular Technology Conference*, vol. 48, pp. 1527–1545, September 1999.

[318] A. J. Goldsmith and S. Chua, "Variable-rate variable-power MQAM for fading channels," *IEEE Transactions on Communications*, vol. 45, pp. 1218–1230, October 1997.

[319] H. Matsuoka and S. Sampei and N. Morinaga and Y. Kamio, "Adaptive Modulation System with Variable Coding Rate Concatenated Code for High Quality Multi-Media Communications Systems," in *Proceedings of IEEE VTC'96*, vol. 1, (Atlanta, USA), pp. 487–491, IEEE, 28 April–1 May 1996.

[320] A. J. Goldsmith and S. Chua, "Adaptive Coded Modulation for Fading Channels," *IEEE Transactions on Communications*, vol. 46, pp. 595–602, May 1998.

[321] C. H. Wong, T. H. Liew and L. Hanzo, "Burst-by-Burst Turbo Coded Wideband Adaptive Modulation with Blind Modem Mode Detection," in *Proceedings of 4th ACTS Mobile Communications Summit 1999*, (Sorrento, Italy), pp. 303–308, June 1999.

[322] D. Goeckel, "Adaptive Coding for Fading Channels using Outdated Fading Estimates," *IEEE Transactions on Communications*, vol. 47, pp. 844–855, June 1999.

[323] B. J. Choi, M. Münster, L. L. Yang, and L. Hanzo, "Performance of Rake receiver assisted adaptive-modulation based CDMA over frequency selective slow Rayleigh fading channel," *Electronics Letters*, vol. 37, pp. 247–249, February 2001.

[324] A. Duel-Hallen, S. Hu, and H. Hallen, "Long Range Prediction of Fading Signals," *IEEE Signal Processing Magazine*, vol. 17, pp. 62–75, May 2000.

[325] S. M. Alamouti and S. Kallel, "Adaptive Trellis-Coded Multiple-Phased-Shift Keying Rayleigh Fading Channels," *IEEE Transactions on Communications*, vol. 42, pp. 2305–2341, June 1994.

[326] L. Piazzo and L. Hanzo, "TTCM-OFDM over Dispersive Fading Channels," *IEEE Vehicular Technology Conference*, vol. 1, pp. 66–70, May 2000.

[327] R. W. Chang, "Synthesis of Band-Limited Orthogonal Signals for Multichannel Data Transmission," *Bell Systems Technical Journal*, vol. 46, pp. 1775–1796, December 1966.

[328] M. S. Zimmermann and A. L. Kirsch, "The AN/GSC-10/KATHRYN/Variable Rate Data Modem for HF Radio," *IEEE Transactions on Communication Technology*, vol. CCM–15, pp. 197–205, April 1967.

[329] L.J. Cimini, "Analysis and Simulation of a Digital Mobile Channel Using Orthogonal Frequency Division Multiplexing," *IEEE Transactions on Communications*, vol. 33, pp. 665–675, July 1985.

[330] F. Mueller-Roemer, "Directions in audio broadcasting," *Journal of the Audio Engineering Society*, vol. 41, pp. 158–173, March 1993.

[331] M. Alard and R. Lassalle, "Principles of modulation and channel coding for digital broadcasting for mobile receivers," *EBU Review, Technical No. 224*, pp. 47–69, August 1987.

[332] B. L. Yeap, T. H. Liew, and L. Hanzo, "Turbo Equalization of Serially Concatenated Systematic Convolutional Codes and Systematic Space Time Trellis Codes," *IEEE Vehicular Technology Conference*, p. 119 (CDROM), May 2001.

[333] ETSI, *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television*, August 1997. ETS 300 744.

[334] U. Wachsmann, R. F. H. Fischer and J. B. Huber, "Multilevel Codes: Theoretical Concepts and Practical Design Rules," *IEEE Transactions on Information Theory*, vol. 45, pp. 1361–1391, July 1999.

[335] M. Isaka and H. Imai, "On the Iterative Decoding of Multilevel Codes," *IEEE Journal on Selected Areas in Comms*, vol. 19, pp. 935–943, May 2001.

[336] S. Lin and D. J. Costello Jr, *Error Control Coding: Second Edition*. USA: Prentice Hall, 2003.

[337] D-F. Yuan, Z-W. Li and F. Zhang, "Two-Level Unequal Error Protection Scheme in Image Transmission System Using Multilevel Codes in Rayleigh Fading Channel," in *IEEE Military Communications Conference (MILCOM)*, vol. 2, pp. 1445 – 1449, October 2001.

[338] M. Isaka, H. Imai, R. H. Morelos-Zaragoza, M. P. C. Fossorier and S. Lin, "Multilevel Codes and Multistage Decoding for Unequal Error Protection," *IEEE Conference for Personal Wireless Communication*, pp. 249–254, February 1999.

[339] R. Gallager, *Information Theory and Reliable Communication*. New York: Willey, 1968.

[340] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes," *Proceedings of the International Conference on Communications*, pp. 1064–1070s, May 1993.

[341] D. F. Yuan, P. Zhang, Q. Wang and W. E. Stark, "A Novel Multilevel Codes With 16QAM," *IEEE Wireless Communications and Networking Conference*, pp. 260–263, March 2002.

[342] L. Papke and K. Fazel, "Combined Multilevel Turbo-code with MR-modulation," in *IEEE International Conference on Communication*, vol. 2, pp. 668–672, June 1995.

[343] L. Papke and K. Fazel, "Combined Multilevel Turbo-code with 8PSK Modulation," in *IEEE Global Communications Conference (GLOBECOM)*, vol. 1, pp. 649–653, November 1995.

[344] T. Woerz and J. Hagenauer, "Iterative Decoding for Multilevel Codes Using Reliability Information," in *IEEE Global Communications Conference (GLOBECOM)*, vol. 3, pp. 1779–1784, December 1992.

[345] R. Bose and D. R.-Chaudhuri, "On a Class of Error Correcting Binary Group Codes," *Information and Control*, vol. 3, pp. 68–79, 1960.

[346] A. Hocquenghem, "Codes correcteurs d'erreurs," *Chiffres (Paris)*, vol. 2, pp. 147–156, September 1959.

[347] R. Gallager, *Low Density Parity Check Codes*. USA: MIT Press, 1963.

[348] L. R. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Transactions on Information Theory*, vol. 20, pp. 284–287, March 1974.

[349] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Canada: John Wiley and Sons Inc., 1991.

[350] U. Wachsmann and J. Huber, "Power and Bandwidth Efficient Digital Communication Using Turbo Codes in Multilevel Codes," *European Transactions on Telecommunications (ETT)*, vol. 6, pp. 557–567, September 1995.

[351] R. Pellizzoni, A. Sandri, A. Spalvieri and E. Biglieri, "Analysis and Implementation of An Adjustable-Rate Multilevel Coded Modulation System," *IEE Proceedings of Communication*, vol. 144, pp. 1–5, February 1997.

[352] S. Lin and P. S. Yu, "A Hybrid ARQ Scheme with Parity Retransmission for Error Control of Satellite Channels," *IEEE Transactions on Communications*, vol. COM-31, pp. 1701–1719, July 1982.

[353] J.-F. Cheng, "Coding Performance of Hybrid ARQ Schemes," *IEEE Transactions on Communications*, vol. 54, pp. 574–574, March 2006.

[354] Q. Huang, S. Chan, L. Ping and K.-T. Ko, "Performance of Hybrid ARQ Using Trellis-Coded Modulation Over Rayleigh Fading Channel," *IEEE Transactions on Vehicular Technology*, vol. 56, pp. 2784–2790, September 2007.

[355] J. Wang, N. S. Othman, J. Kliewer, L. L. Yang and L. Hanzo, "Turbo-Detected Unequal Error Protection Irregular Convolutional Codes Designed For The Wideband Advanced Multirate Speech Codec," *IEEE Transactions on Vehicular Technology*, vol. 2, pp. 927 – 931, September 2005.

[356] L. Hanzo, T. H. Liew and B. L. Yeap, *Turbo Coding, Turbo Equalisation and Space Time Coding for Transmission over Wireless channels*. New York, USA: John Wiley IEEE Press, 2002.

[357] N. H. Tran and H. H. Nguyen, "Signal Mappings of 8-Ary Constellations for Bit Interleaved Coded Modulation With Iterative Decoding," *IEEE Transactions on Broadcasting*, vol. 52, pp. 92–99, March 2006.

[358] S. T. Brink, J. Speidel and R-H. Yan, "Iterative Demapping and Decoding for Multilevel Modulation," in *IEEE Global Communications Conference (GLOBECOM)*, vol. 1, pp. 579 – 584, November 1998.

[359] S. T. Brink, "Convergence Behavior of Iteratively Decoded Parallel Concatenated Codes," *IEEE Transactions on Communications*, pp. 1727–1737, October 2001.

[360] F. Schreckenbach and G. Bauch, "EXIT charts for iteratively decoded multilevel modulation," *12th European Signal Processing Conference (EUSIPCO)*, September 2004.

[361] D. Divsalar, S. Dolinar and F. Pollara, "Serial Concatenated Trellis Coded Modulation with Rate-1 Inner Code," *IEEE Global Telecommunications Conference*, pp. 777–782, November 2000.

[362] Y. Yasuda, K. Kashiki and Y. Hirata, "High-Rate Punctured Convolutional Codes for Soft Decision Viterbi Decoding," *IEEE Transactions on Communications*, pp. 315–319, March 1984.

[363] S. Lin and D. J. Costello, Jr, *Error Control Coding: Fundamentals and Applications*. Inc. Englewood Cliffs, New Jersey 07632: Prentice-Hall, 1983.

[364] J. G. Proakis, *Digital Communications*. Mc-Graw Hill International Editions, 3rd ed., 1995.

[365] Y. Bian, A. Popplewell and J. J. O'Reilly, "New Very High Rate Punctured Convolutional Codes," *IEE Electronics Letters*, vol. 30, pp. 1119–1120, July 1994.

[366] L. Hanzo, C. Somerville, and J. Woodard, *Voice and Audio Compression for Wireless Communications*. New York, USA: John Wiley IEEE Press, 2007.

[367] L. Hanzo, P.J. Cherriman and J. Street, *Video Compression and Communications: From Basics to H.261, H.263, H.264, MPEG4 for DVB and HSDPA-Style Adaptive Turbo-Transceivers*. Chichester, England: John Wiley and Sons Ltd, 2007.

[368] S. T. Brink, "Convergence of Iterative Decoding," *IEE Electonics Letter*, vol. 35, pp. 806–808, May 1999.

[369] F. Brännström, L. K. Rasmussen and A. J. Grant, "Convergence Analysis and Optimal Scheduling for Multiple Concatenated Codes," *IEEE Transactions on Information Theory*, vol. 51, pp. 3354–3364, September 2005.

[370] J. Hagenauer, "The EXIT Chart - Introduction To Extrinsic Information Transfer In Iterative Processing," *European Signal Processing Conference*, pp. 1541–1548, September 2004.

[371] M. Tüchler, "Convergence prediction for iterative decoding of threefold concatenated systems," *IEEE Global Telecommunications Conference*, pp. 1358–1362, Nov. 2002.

[372] F. Brännström, L. K. Rasmussen and A. Grant, "Optimal Scheduling for Iterative Decoding," *IEEE International Symposium on Information Theory*, p. 350, June 2003.

[373] K. R. Narayanan, "Effect of Precoding on the Convergence of Turbo Equalization for Partial Response Channels," *IEEE Journal on Selected Areas in Communications*, pp. 686–698, Apr. 2001.

[374] I. Lee, "The Effect of a Precoder on Serially Concatenated Coding Systems With an ISI Channel," *IEEE Transaction On Communications*, vol. 49, pp. 1168–1175, July 2001.

[375] A. G. Lillie, A. R. Nix and J. P. McGeehan, "Performance and Design of a Reduced Complexity Iterative Equalizer for Precoded ISI Channels," *Vehicular Technology Conference (VTC)*, pp. 299–303, October 2003.

[376] N. H. Tran and H. H. Nguyen, "Signal Mappings of 8-ary Constellations for BICM-ID Systems Over a Rayleigh Fading Channel," *IEEE Communications Letters*, vol. E88-B, pp. 4083–4086, October 2005.

[377] F. Shreckenbach, N. Görtz, J. Hagenauer and G. Bauch, "Optimization of Symbol Mappings for Bit-Interleaved Coded Modulation With Iterative Decoding," *IEEE Communication Letters*, pp. 593–595, December 2003.

[378] F. Simoens, H. Wymeersch and M. Moeneclaey, "Multi-dimensional Mapping for Bit-Interleaved Coded Modulation," *IEEE Vehicular Technology Conference-Spring (VTC)*, pp. 733–737, May 2005.

[379] D-F. Yuan, F. Zhang, A-F. So, Z-W. Li, "Concatenation of Space-Time Block Codes and Multilevel Coding over Rayleigh Fading Channels," in *IEEE Vehicular Technology Conference*, (Atlantic City, USA), pp. 192–196, October Fall 2001.

[380] S. M. Alamouti, "A Simple Transmitter Diversity Scheme for Wireless Communications," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 1451–1458, October 1998.

[381] V. Tarokh, H. Jafarkhani and A.R. Calderbank, "Space-Time Block Codes from Orthogonal Designs," *IEEE Transactions on Information Theory*, vol. 45, pp. 1456–1467, July 1999.

[382] W. Su, Z. Safar, and K. J. R. Liu, "Space-Time Signal Design for Time-correlated Rayleigh Fading Channels," in *IEEE International Conference on Communications*, (Anchorage, Alaska), pp. 3175–3179, May 2003.

[383] O. Alamri, B. L. Yeap, L. Hanzo, "Turbo Detection of Channel-Coded Space-Time Signals Using Sphere Packing Modulation," in *IEEE Vehicular Technology Conference*, (Los Angeles, USA), pp. 2498–2502, September Fall 2004.

[384] O. Alamri, B. L. Yeap and L. Hanzo, "A Turbo Detection and Sphere-Packing-Modulation Aided Space-Time Coding Scheme," *IEEE Transactions on Vehicular Technology*, vol. 56, pp. 575–582, March 2007.

[385] J. H. Conway and N. J. Sloane, "Sphere Packings, Lattices and Groups," *Spinger-Verlag*, 1999.

[386] O. Alamri, B. L. Yeap, L. Hanzo, "Turbo Detection of Channel-Coded Space-Time Signals Using Sphere Packing Modulation," in *IEEE Vehicular Technology Conference*, (Los Angeles, USA), pp. 2498–2502, September Fall 2004.

[387] O. R. Alamri, *Turbo Detection of Sphere Packing Modulation Aided Space-Time Coding Schemes*. PhD , University of Southampton, 2007.

[388] V. Tarokh, N. Seshadri and A. R. Calderbank, "Space-time Codes for High Rate Wireless Communication: Performance analysis and code construction," *IEEE Transactions on Information Theory*, vol. 44, pp. 744–765, March 1998.

[389] G. H. Golub and C. F. Van-Loan, *Matrix Computations*. Baltimore, MD: Johns Hopkins, 1996.

[390] S. X. Ng and L. Hanzo, "On the MIMO Channel Capacity of Multi-Dimensional Signal Sets," *IEEE Transactions on Vehicular Technology*, vol. 55, pp. 528–536, March 2006.

[391] L. He and H. Ge, "A New Full-Rate Full-Diversity Orthogonal Space-Time Block Coding Scheme," *IEEE Communications Letters*, vol. 7, pp. 590–592, December 2003.

[392] K. Zeger and A. Gersho, "Pseudo-Gray Coding," *IEEE Transaction On Communications*, vol. 38, pp. 2147–2158, December 1990.

[393] M. Isaka, R. H. M-Zaragoza, M. P. C. Fossorier, S. Lin and H. Imai, "Multilevel Coded 16-QAM Modulation with Multistage Decoding and Unequal Error Protection," in *Global Communication Conference*, vol. 6, (Sydney, Austrialia), pp. 3548–3553, November 1998.

[394] F. Guo, "Low Density Parity Check Coding," *Ph.D thesis, University of Southampton,UK*, 2005.

[395] Y. Huang and J. A. Ritcey, "Improved 16-QAM Constellation Labeling for BI-STCM-ID With the Alamouti Scheme," *IEEE Communications Letters*, vol. 9, pp. 157–159, February 2005.

[396] J. Hou. P. H. Siegel, L. B. Milstein and H. D. Pfister, "Multilevel Coding with Low-Density Parity-Check Component Codes," *IEEE Global Telecommunications Conference*, pp. 1016–1020, Nov. 2001.

[397] J. Boutros, O. Pothier and G. Zḿor, "Generalized Low Density (Tanner) Codes," *Proceedings of the International Conference of Communication*, pp. 441–445, July 1999.

[398] M. Lentmaier and K. S. Zigangirov, "On Generalized Low-Density Parity-Check Codes Based on Hamming Component Codes," *IEEE Communications Letters*, pp. 248–250, August 1999.

[399] Bernard Sklar, *Digital communications: Fundamentals and Applications*. Inc. Englewood Cliffs, New Jersey 07632: Prentice-Hall, 1988.

[400] D. J. C. MacKay, *Fountain Codes*. http://www.inference.phy.cam.ac.uk/mackay/CodesTheory.html: Cavendish Laboratory, University of Cambridge, 2004.

[401] D. J. C. MacKay, *Information Theory, Interference and Learning Algorithms*. http://www.inference.phy.cam.ac.uk/mackay/itila/: Cambridge University Press, 2003.

[402] M. Luby, "LT codes," in *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*, pp. 271–282, November 2002.

[403] R. M. Tanner, "A Recursive Approach To Low Complexity Codes," *IEEE Transaction On Information Theory*, vol. 27, pp. 533–547, September 1981.

[404] O. Pothier, *Compound Codes Based On Graphs and Their Iterative Decoding*. PhD, Ecole Nationale Suprieure des Telecommunications, January 2000.

[405] J. Buyers, M. Luby, M. Mitzenmacher and A. Rege, "A digital Fountain approach to reliable distribution of bulk data," *In Proceedings of ACM SIGCOMM'98*, September 1998.

[406] M. G. Luby, M. Shokrollahi, M. Watson and T. Stockhammer, "Raptor Forward Error Correction Scheme," *Reliable Multicast Transport Internet-Draft*, June 2005. URL available at :*http://www.rfc-archive.org/getrfc.php?rfc=5053*.

[407] D. J. C. MacKay, "Fountain Codes," *IEEE Proceedings in Communications*, pp. 1062–1068, December 2005.

[408] T. D. Nguyen, L. L. Yang and L. Hanzo, "Systematic Luby Transform Codes and Their Soft Decoding," *IEEE Workshop on Signal Processing Systems (SiPS)*, pp. 67–72, October 2007.

[409] L. Hanzo, L-L. Yang, E-L. Kuan and K. Yen, *Single- and Multi-Carrier DS-CDMA*. New York, USA: John Wiley IEEE Press, 2003.

[410] A. Shokrollahi, "Raptor Codes," *International Symposium on Information Theory*, p. 36, June 2004.

[411] A. Shokrollahi, *Raptor Codes*. http://algo.epfl.ch: Technical Report, June 2003.

[412] J. Buyers, M. Luby and M. Mitzenmacher, "A Digital Fountain Approach to Asynchronous Reliable Multicast," *IEEE Journal on Selected Areas in Communications*, vol. 20, pp. 1528–1540, October 2002.

[413] Q. Luo and P. Sweeney, "Hybrid-ARQ Protocols Based on Multilevel Coded Modulation," *IEE Electronics Letters*, vol. 39, pp. 1063 – 1065, July 2003.

[414] T. Clevorn, S. Godtmann and P. Vary, "BER Prediction Using EXIT Charts For BICM With Iterative Decoding," *IEEE Communications Letters*, vol. 10, pp. 49–51, January 2006.

[415] N. H. Tran and H. H. Nguyen, "Signal Mappings of 8-Ary Constellations for BICM-ID Systems over a Rayleigh Fading Channel," *IEICE Transcations Letter on Communication*, vol. E88-B, pp. 4083–4086, October 2005.

[416] F. Simoens, H. Wymeersch and M. Moeneclaey, "Multi-dimensional Mapping for Bit-Interleaved Coded Modulation," *Vehicular Technology Conference, Spring*, vol. 2, pp. 733–737, June 2005.

[417] F. Schreckenbach and G. Bauch, “Bit-Interleaved Coded Irregular Modulation,” *European Transactions on Telecommunications*, vol. 7, pp. 269–282, April 2006.

[418] L. Hanzo, C. H. Wong, and M. S. Yee, *Adaptive Wireless Transceivers: TurboCoded, TurboEqualized and Space-Time Coded TDMA, CDMA, and OFDM Systems*. Chichester, UK: John Wiley IEEE Press, August 2002.

[419] F. Simoens, H. Wymeersch and M. Moeneclaey, “Design and Analysis of Linear Precoders for Bit-Interleaved Coded Modulation with Iterative Decoding,” in *International Symposium on Turbo Codes and Related Topics*, (Munich, Germany), April 2006.

[420] L. Szczecinski, H. Chafnaji and C. Hermosilla, “Modulation Doping for Iterative Demapping of Bit-Interleaved Coded Modulation,” *IEEE Communications Letters*, vol. 9, pp. 1031–1033, December 2005.

[421] M. Tüchler, “Design of Serially Concatenated Systems Depending on The Block Length,” *IEEE Transactions on Communications*, vol. 52, pp. 209–218, February 2004.

[422] M. Tüchler and J. Hagenauer, “EXIT Charts of Irregular Codes,” *IEEE Conference on Information Sciences and Systems*, pp. 748–753, March 2002.

[423] A. Ashikhmin, G. Kramer and S. T. Brink, “Extrinsic Information Transfer Functions: Model and Erasure Channel Properties,” *IEEE Transactions on Information Theory*, vol. 50, pp. 2657–2673, November 2004.

[424] A. Ashikhmin, G. Kramer and S. T. Brink, “Code Rate and the Area under Extrinsic Information Transfer Curves,” *IEEE International Symposium on Information Theory*, p. 115, June 2002.

[425] R. G. Maunder and L. Hanzo, “Near-capacity irregular variable length coding and irregular unity rate coding,” IEEE Transactions on Wireless Communications*, available at :*http://eprints.ecs.soton.ac.uk/14471/, 2008.

[426] L. Hanzo, S. X. Ng, T. Keller and W. Webb, *Quadrature Amplitude Modulation*. Chichester, UK : John Wiley and Sons, 2004.

[427] T. H. Liew, B. L. Yeap, J. P. Woodard, and L. Hanzo, “Modified MAP Algorithm for Extended Turbo BCH Codes and Turbo Equalisers,” in *Proceedings of the First International Conference on 3G Mobile Communication Technologies, January 2000*, (London, UK), pp. 185–189, 27-29 March 2000.

[428] L. Zhao, L. Lampe and J. Huber, “Study of Bit-Interleaved Coded Space-Time Modulation with Different Labeling,” *IEEE Information Theory Workshop*, pp. 199–202, March 2003.

[429] D. Chase, “A Combined Coding and Modulation Approach for Communication over Dispersive Channels,” *IEEE Transactions on Communications*, vol. COM-21, pp. 159 – 173, March 1973.

[430] B. A. Harvey and S. B. Wicker, “Packet combining systems based on the Viterbi decoder,” *IEEE Transactions on Communications*, vol. 42, pp. 1544 – 1557, March 1994.

[431] L. Y. Song and A. G. Burr, “A Simple Differential Modulation Scheme for Quasi-Orthogonal Space-Time Block Codes with Partial Transmit Diversity,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2007, pp. 13–19, January 2007.

[432] W. Lee, “Estimate of channel capacity in Rayleigh fading environment,” *IEEE Transactions on Vehicular Technology*, vol. 39, pp. 187–189, August 1990.

[433] S. Nanda, K. Balachandran, and S. Kumar, “Adaptation techniques in wireless packet data services,” *IEEE Communications Magazine*, vol. 38, pp. 54–64, January 2000.

[434] L. Hanzo, P. Cherriman, and J. Streit, *Wireless Video Communications: From Second to Third Generation Systems, WLANs and Beyond*. Piscataway, NJ, USA: IEEE Press, 2001. (For detailed contents please refer to http://www-mobile.ecs.soton.ac.uk).

[435] L. Hanzo, C. Wong, and P. Cherriman, “Channel-adaptive wideband video telephony,” *IEEE Signal Processing Magazine*, vol. 17, pp. 10–30, July 2000.

[436] L. Hanzo, P. Cherriman, and E. Kuan, “Interactive cellular and cordless video telephony: State-of-the-art, system design principles and expected performance,” *Proceedings of IEEE*, September 2000.