Université catholique de Louvain
Ecole Polytechnique de Louvain
Laboratoire de Télécommunications et Télédétection

Coded Transmission Systems for the Orthogonal Relay Channel

Thesis presented by Harold H. Sneessens
for the Ph.D. degree in Applied Sciences.

Examining Committee :

Luc Vandendorpe, supervisor	Université catholique de Louvain
Jean-Pierre Raskin, chairman	Université catholique de Louvain
Joachim Hagenauer	Technische Universität München
J. Nicholas Laneman	University of Notre Dame
Jérôme Louveaux	Université catholique de Louvain
Marc Moeneclaey	Universiteit Gent
Claude Oestges	Université catholique de Louvain

July 2009
Louvain-la-Neuve, Belgium.

Acknowledgments

As I reach the end of my Ph.D. thesis, I feel deep gratitude towards my advisor, Pr. Luc Vandendorpe, for his support during all these years. His teaching skills arose my interest for telecommunications, and were a decisive factor for starting the Ph.D. All along the way, he provided me with a very supportive environment to make progress in my work.

I also thank Pr. J. Nicholas Laneman, who has been so kind as to let me come and work at the University of Notre Dame. His thorough knowledge of my thesis subject and his remarkably great insight into telecommunications in general have been quite simply invaluable. I especially want to thank him for making these interactions so enjoyable, and I look forward to developing these further.

I also enjoyed collaborations with my co-authors, Cédric Herzet and Xavier Jaspar, whose insight has significantly contributed to develop my work. Thank you both for your help along the way.

Work would never have been so pleasant if our laboratory had not been full of great people. To you all, thanks for having had so much fun beyond work. With some of you we used to stretch days far beyond sunset : I always look at this extra life time with fond memories. I look forward to writing the continuation of this story.

Last but not least (by a long way), I want to express my deep appreciation to all my family and friends, whose continued support is so crucial. A short mention here cannot adequately capture all my gratitude towards you. I hope I will be able to give back even more than I received from each of you — I am eager to do so.

Abstract

Communication over wireless channels is affected by fading, i.e., fast random variations of the received signal strength. To prevent fading from causing communication outages, a receiver can acquire several copies of the signal and recombine them to average individual variations. In cooperative networks, terminals achieve this by propagating the signal along multiple paths from its source to its destination.

The orthogonal relay channel is a simple model for such cooperative transmission, in which a relay helps a source and a destination communicate. Protocols for relaying are separated into three categories, depending on the kind of processing at the relay. In Decode-Forward (DF) protocols, the relay decodes, re-encodes and forwards the signal it receives. In Amplify-Forward (AF) protocols, the relay simply amplifies the signal before forwarding it. And in Compress-Forward (CF) protocols, the relay forwards a compressed version of the signal it observes.

This thesis is concerned with the design and analysis of relaying protocols, with a focus on cases where the relay is not able to decode the signal perfectly.

A theoretical analysis of the CF protocol shows that the optimal compression method, Wyner-Ziv coding, often reduces to simpler rate-distortion coding when the relay has not enough information on the state of the channels.

Practical implementations of relaying protocols follow. The first implementation, Soft Decode-Forward, is a DF relay that forwards the data with an amplitude reflecting its reliability. The second implementation is a receiver for the DF protocol, that takes into account the uncertainty at the relay. The third is a receiver for the CF protocol that jointly decodes the signals from the source and from the relay. Comparisons show that all these implementations outperform usual protocols. In hybrid schemes over wireless channels, the CF protocol with joint reception is the most promising alternative.

Contents

Notations and acronyms

NOTATIONS

Random variables : x is a realization of the random variable X; the probability $P(x)$ is the probability density/mass function of X evaluated in $X = x$. The notation $X - Y - Z$ indicates that X, Y and Z form a Markov chain. The expected value of X is written $\mathrm{E}\{X\}$. The abbreviation *i.i.d.* stands for *independent and identically distributed*. By convention, in this document binary variables take values in $\{-1, +1\}$.

Vectors and matrices : Vectors are represented with bold lowercase letters (e.g. $\boldsymbol{x}$), and matrices with bold uppercase letters (e.g. $\boldsymbol{M}$). Vectors and matrices are indexed with square brackets : $\boldsymbol{x}[n]$ is the n-th element of the vector $\boldsymbol{x}$, and $\boldsymbol{M}[m, n]$ is the element in row m, column n of $\boldsymbol{M}$. The determinant of $\boldsymbol{M}$ is written $|\boldsymbol{M}|$.

Miscellaneous : The absolute value or modulus of a is written $|a|$. The conjugate of a complex number a is written a^*. The marginal of the function $f(x, \ldots)$ with respect to x, i.e. the summation over all variables except x, is written $\sum_{\sim\{x\}} f(x, \ldots)$. The function $\mathbb{I}\{\text{statement}\}$ is equal to 1 if the statement is true, and to 0 otherwise. The sign $\triangleq$ means *is defined as*, the sign $\propto$ means *is proportional to* and the sign $\approx$ means *is approximately equal to*.

ACRONYMS

AF	Amplify-Forward
ASK	Amplitude-Shift Keying
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BCJR	Bahl Cocke Jelinek Raviv
BICM	Bit-Interleaved Coded Modulation
BPSK	Binary Phase Shift Keying
CF	Compress-Forward
CRC	Cyclic Redundancy Check
CSI	Channel State Information
DF	Decode-Forward
EXIT	EXtrinsic Information Transfer
FER	Frame Error Rate
LDPC	Low-Density Parity Check
LLR	Log-Likelihood Ratio
MAP	Maximum A Posteriori
MRC	Maximum Ratio Combining
NSC	Non-recursive Systematic Convolutional
RSC	Recursive Systematic Convolutional
SISO	Soft-Input Soft-Output
SNR	Signal-to-Noise Ratio

Introduction

The future has come, as wireless communication technologies have invaded our lives. Your pocket certainly contains a phone, maybe a television, and perhaps even a computer connected to information networks. In less than a decade, wireless communications devices and networks went from esoteric science-fiction to ubiquitous mainstream culture. Far from being confined to gadgets, they even became of critical social and economical importance.

The rapid growth of telecommunication networks has been supported by large amounts of research. However today, as wireless broadband data services are making inroads into cellular networks, many challenges remain open to support further developments and improvements of these services.

Wireless communications technologies are not a straightforward extension of their wired counterparts. Differences are to a large extent due to the properties of electromagnetic propagation in the wireless medium, and to the network topologies resulting thereof.

Electro-magnetic propagation in the wireless medium is a vast subject, but three properties are particularly important to motivate this work. Firstly, wireless transmissions are broadcast by nature : signals are transmitted over the air, and can thus be heard by any terminal within the transmission range. Secondly, electro-magnetic signals are subject to reflection, refraction and diffusion by scattering objects of the environment, as are for example buildings or people. As a result, a signal takes multiple paths between its source and its destination. Such multi-path propagation has advantages and drawbacks. An advantage is that communication is possible even without line-of-sight between the source and destination terminals, since signals can reach their destination after several reflections or diffractions. However, a significant drawback of multi-path propagation is that these multiple copies of the signal interfere at the destination, con-

structively or destructively, thereby causing variations of the received signal strength. Such random time and space variations of the received signal strength are referred to as fading. Finally, a third important property of wireless propagation is that signals are subject to path-loss, i.e., strong attenuation as they propagate through space.

The problem of communicating on such channels suggests part of its solution. Due to the broadcast nature of wireless transmission, wireless networks are most often highly connected since terminals in a same neighborhood can easily hear each other. This is even more true given the often high density of terminals in the environment, as is the case for instance in cellular networks and in most wireless computer networks. It turns out that using this highly-connected network topology to get signals across the network is an efficient way to overcome fading and path-loss. This is exactly the idea behind cooperative communications, in which users of a wireless network can help each other to get their messages to their destination.

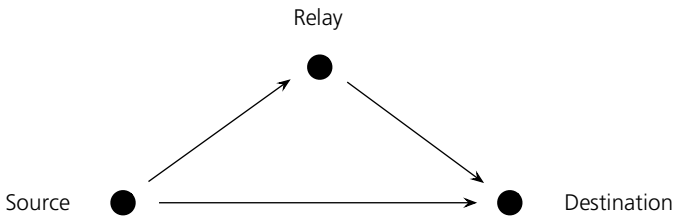


Figure 1: Relay channel.

As a simple example of cooperative communications, consider the so-called relay channel of fig. 1. In this network, a single relay helps a pair of terminals communicate. An obvious gain in using such a relay is that, due to the nature of path-loss, it is much more energy-efficient to transmit the signal over two small hops, from the source to the relay and from the relay to the destination, than directly from the source to the destination. An equally interesting gain is that the destination receives two redundant copies of the signal, so that the probability that both experience a deep fade is drastically reduced. This is referred to as a diversity advantage.

However, what a relaying terminal should do to help two other terminals communicate is not necessarily obvious. For this reason, and given the prospects of cooperation, protocols for communicating over the elementary model of fig. 1 currently receive a lot of attention. Such protocols are usually classified into three categories. In the

first category of protocols, called Decode-Forward (DF), the relay decodes, re-encodes and forwards the signal to the destination. In protocols of the second family, Amplify-Forward (AF), the relay simply amplifies the signal it receives and forwards it to the destination. And finally, in the third family of protocols termed Compress-Forward (CF), the relay compresses the signal it receives and sends it to the destination.

This thesis is concerned with the design and analysis of these protocols, especially for cases where the relay is not able to decode the signal perfectly.

From a theoretical point of view, this thesis analyses the nature of compression in the CF protocol, when the transmitters have no information on the state of the channel. It shows that, in that case, the optimal compression method, Wyner-Ziv coding or rate-distortion coding with side information, often reduces to usual rate-distortion coding.

From a more practical point of view, this thesis presents three techniques for dealing with signals known with limited reliability at the relay. Throughout the work, the intention has been to include the codes in the design, as their use has a significant impact on the nature of the protocols. A first technique, termed Soft Decode-Forward, resembles a DF system where all operations are performed in a soft fashion; its aim is to regenerate the signal while keeping reliability information on the decoded data. A second technique is a receiver for the DF protocol that manages to handle errors made by the relay. And finally, a third technique is an iterative receiver for the CF protocol. Differing from usual CF receivers, this receiver decodes jointly the signals received from the source terminal and from the relay. This enables to benefit from the error-correcting codes to enhance the side-information used for Wyner-Ziv decoding. These three techniques are then compared in the context of hybrid protocols. The CF protocol with iterative reception is identified as the most interesting of all three alternatives, in most situations.

Outline and contributions

Chapter 1 – Factor graphs presents the factor graphs framework, used in later chapters to design several communication devices.

Chapter 2 – Codes and factor graphs sets the problem of receiver design in the factor graphs framework. Decoders are derived for codes used in this thesis : convolutional codes, turbo-codes, and bit-interleaved coded modulation. The chapter ends with a presentation of extrinsic information transfer charts, a tool to analyze iterative

receivers, and with a derivation of an analytical method to analyze the convergence of the turbo-decoding algorithm.

This method for analytical analysis of the convergence of the turbo-decoding algorithm is an original contribution of this thesis (first reported in [1, 2]).

Chapter 3 – Relaying on orthogonal channels introduces the orthogonal relay channel along with the DF, AF and CF protocols in an information-theoretic context.

Chapter 4 – Adaptive Compress-Forward relaying in fading environments focusses on the CF protocol over wireless channels, when the terminals have channel state information at the receiver only. It shows that the optimal compression method is often conventional source compression instead of Wyner-Ziv coding.

This chapter is an original contribution of this thesis (first reported in [3, 4]).

Chapter 5 – Practical codes and decoders for the relay channel deals with implementable coded systems for the relay channel. After a presentation of a DF system and of an AF system, it introduces three techniques for transmitting signals with limited reliability at the relay : the Soft Decode-Forward relaying technique, a receiver for the DF protocols that handles errors made by the relay, and an iterative receiver for the CF protocol. These protocols are then compared in hybrid transmission schemes that use DF if the relay is able to decode, and one of the three protocols above otherwise. This enables to identify CF with iterative reception as a very promising alternative.

The Soft-DF protocol, the DF receiver resilient to relay errors and the iterative receiver for the CF protocol are original contributions of this thesis (first reported in [5, 6, 7, 8, 9]).

Factor graphs

1

The computation of the marginals of a global function can be simplified by exploiting how this global function factorizes into a product of local functions.

This chapter presents a generic algorithm taking advantage of such a factorization. This algorithm, called the *sum-product algorithm*, operates on a *factor graph*, i.e. on a graphical representation of the factorization of the function. The algorithm consists in making messages pass on the factor graph, using simple propagation and processing rules. The sum-product algorithm is strictly valid for acyclic factor graphs only, but it has been successfully used on cyclic graphs as well.

In addition to the sum-product algorithm, this chapter presents the sub-optimal max-sum algorithm, obtained by simplifying the processing of messages.

The general framework of this chapter has applications in several domains, as for example telecommunications, signal processing, or artificial intelligence. It is used in this thesis to design receivers for communication over the relay channel.

This chapter follows closely [10, 11], as well as [12, 13, 14].

Contents

1.1 DEFINITIONS AND NOTATIONS

Consider variables $X_1, \ldots, X_n$ and their domains $\mathcal{A}_1, \ldots, \mathcal{A}_n$. We study functions

$$g : \mathcal{S} \to \mathcal{R}, (X_1, \ldots, X_n) \to g(X_1, \ldots, X_n),$$

where the domain $\mathcal{S}$ is $\mathcal{A}_1 \times \ldots \times \mathcal{A}_n$ and where the codomain $\mathcal{R}$ can in general be any semiring. For the applications we consider, taking $\mathcal{R} = \mathbb{R}$, the set of real numbers, suffices.

1.1.1 Marginals of a function

To each function $g(X_1, \ldots, X_n)$, we associate the n marginal functions $g_i(X_i)$, ($i = 1, \ldots, n$):

$$g_i(a) = \sum_{\substack{(X_1,\ldots,X_n)\in\mathcal{S} \\ X_i=a}} g(X_1, \ldots, X_n) \qquad \forall a \in \mathcal{A}_i. \tag{1.1}$$

To represent the marginalization with respect to a variable, we define the shorthand notation of "not-sum" or "summary". Rather than writing the variables being summed over, we write those not being summed over. Equation (1.1) expressing the marginal function of g relative to X_i becomes with this convention

$$g_i(a) = \sum_{\sim\{X_i\}} g(X_1, \ldots, X_n). \tag{1.2}$$

This notation is used throughout this document.

1.1.2 Factor graphs

The calculation of the marginals of a function can be simplified by taking advantage of the way this function factorizes. To this end, a useful tool is the factor graph of the function, i.e., a graph that represents the factors of the function and their dependencies with the variables.

A function $g(X_1, \ldots, X_n)$ can in general be factorized into a product of local functions depending on a subset of the variables $\{X_1, \ldots, X_n\}$. We write such a factorization as follows :

$$g(X_1, \ldots, X_n) = \prod_{\boldsymbol{X}\in\mathcal{Q}} f_{\boldsymbol{X}}(\boldsymbol{X}), \tag{1.3}$$

where $\boldsymbol{X}$ is a subset of the variables $\{X_1, \ldots, X_n\}$, $\mathcal{Q}$ is a collection of subsets of $\{X_1, \ldots, X_n\}$, and $f_{\boldsymbol{X}}(\boldsymbol{X})$ is a function with the elements of $\boldsymbol{X}$ as arguments.

Definition 1 (Factor graph). *A factor graph is a bipartite graph representing the structure of the factorization* (1.3)*. A factor graph has a variable node for each variable X_i and a factor node for each local function $f_{\boldsymbol{X}}(\boldsymbol{X})$. An edge connects the variable node X_i and the factor node $f_{\boldsymbol{X}}(\boldsymbol{X})$ if and only if X_i is an argument of $f_{\boldsymbol{X}}(\boldsymbol{X})$, i.e., if $X_i \in \boldsymbol{X}$.*

Example 1.1 : Factor graph

Consider a function $g(X_1, X_2, X_3, X_4, X_5)$ of five variables, that factorizes into the following product :

$$g(X_1, X_2, X_3, X_4, X_5) = f_A(X_1) f_B(X_2) f_C(X_1, X_2, X_3) f_D(X_3, X_4) f_E(X_3, X_5). \tag{1.4}$$

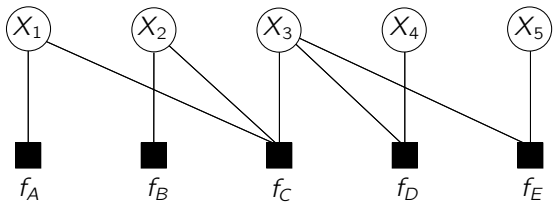


Figure 1.1: Factor graph of $g(X_1, X_2, X_3, X_4, X_5)$.

The corresponding factor graph is represented in 1.1. Note that in this case the graph is a tree. ■

1.2 MARGINALS AND THE SUM-PRODUCT ALGORITHM

Efficient calculation of all the marginals $g_i(X_i)$, $i = 1, \ldots, n$ of a function should reuse as much as possible intermediary calculations made for each marginal.

In this section, we first show how the factor graph of the function enables to identify these common intermediate calculations. Next we present the sum-product algorithm, a generic algorithm using the previous observations to calculate efficiently all the marginals of a function. The sum-product algorithm is then illustrated with an example.

1.2.1 Formal similarities between factor graphs and expression trees

Illustration

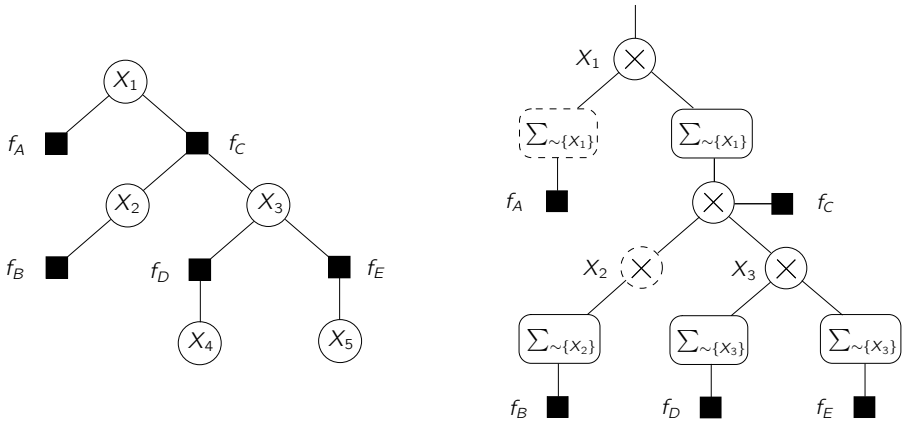
Consider the function of example 1.1. The marginal $g_1(X_1)$ can be written

$$g_1(X_1) = f_A(X_1) \times \left(\sum_{X_2} f_B(X_2) \left(\sum_{X_3} f_C(X_1, X_2, X_3) \left(\sum_{X_4} f_D(X_3, X_4)\right) \left(\sum_{X_5} f_E(X_3, X_5)\right)\right)\right), \tag{1.5}$$

or equivalently, using the summary notation,

$$g_1(X_1) = f_A(X_1) \times \sum_{\sim\{X_1\}} \left(f_B(X_2)\, f_C(X_1, X_2, X_3) \left(\sum_{\sim\{X_3\}} f_D(X_3, X_4)\right) \left(\sum_{\sim\{X_3\}} f_E(X_3, X_5)\right)\right). \tag{1.6}$$

The right-hand member of expression (1.6) can be represented by a so-called *expression tree*, as in fig. 1.2 (b).



(a) (b)

Figure 1.2: (a) Factor graph of example 1.1, represented as a tree with X_1 as root. (b) Expression tree for $g_1(X_1)$. The operands of each operator are its children nodes in the tree.

The comparison between the factor graph and the expression tree in fig. 1.2 shows that they share the same formal structure. Thus, a factor graph not only represents a factorization of a function, it also represents a way to calculate its marginals.

Transformation of a factor graph into an expression tree

The correspondence between a factor graph and the expression tree for the calculation a marginal is clear from fig. 1.2. The following operations convert an acyclic factor graph of a function $g(X_1, \ldots, X_n)$ into an expression tree for $g_i(X_i)$, $i = 1, \ldots, n$:

1. Draw the acyclic factor graph with X_i as root.
2. Replace each variable node by a product operator (fig. 1.3 (a)).
3. Replace each factor node f by a product operator connected to a function node f, then place an operator $\sum_{\sim\{X\}}$ between f and its parent node X (fig. 1.3 (b)).

In some places these operations are trivial and can be omitted. This happens for product operations with less than two operands, and for $\sum_{\sim\{X\}}$ operators that apply to a function depending on X only. These trivial operations are drawn with dashed lines in fig. 1.2.

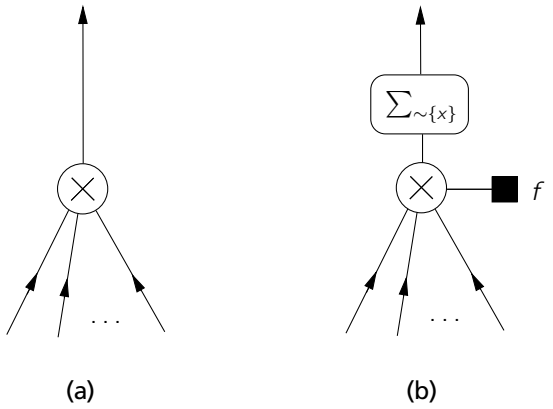


Figure 1.3: Substitutions that transform an acyclic factor graph into an expression tree for the calculation of a marginal : (a) for a variable node; (b) for a factor node.

General case and justification

We now give the justification for the correspondence between the calculation of marginals and the structure of the factor graph, for acyclic graphs.

Consider a function $g(X, X_1, \ldots, X_{n-1})$ that can be represented by an acyclic factor

graph, a factor tree T. We are interested in the marginal with respect to X, i.e.,

$$\sum_{\sim\{X\}} g(X, X_1, \ldots, X_{n-1}). \tag{1.7}$$

We take X as root for T; all other nodes are thus descendants of X.

If X has K neighbors in the tree T (fig. 1.4), we can write g as follows, without loss of generality :

$$g(X, X_1, \ldots, X_{n-1}) = \prod_{k=1}^{K} F_k(X, \boldsymbol{V}_k) \tag{1.8}$$

where $F_k(X, \boldsymbol{V}_k)$ is the product of all local functions that are descendants of the k^{th} neighbor of X, and $\boldsymbol{V}_k$ is the set of all variables that are descendants of the k^{th} neighbor of X. Since T is a tree, $\boldsymbol{V}_1, \ldots, \boldsymbol{V}_{n-1}$ is a partition of $X_1, \ldots, X_{n-1}$: we have $\boldsymbol{V}_i \cap \boldsymbol{V}_j = \emptyset$ for $i \neq j$, and $\boldsymbol{V}_1 \cup \cdots \cup \boldsymbol{V}_{n-1} = \{X_1, \ldots, X_{n-1}\}$. The graph corresponding to this factorization is represented in fig. 1.4, where $F_1(X, \boldsymbol{V}_1)$ is shown factorized.

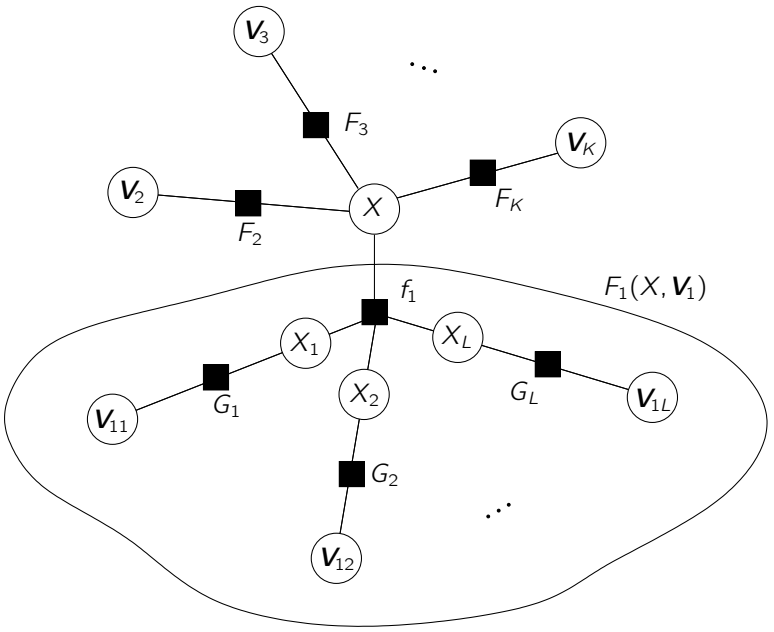


Figure 1.4: Factor graph of $g(X, X_1, \ldots, X_{n-1}) = \prod_{k=1}^{K} F_k(X, \boldsymbol{V}_k)$.

Since $\boldsymbol{V}_1, \ldots, \boldsymbol{V}_K$ are pairwise disjoint, we obtain by the distributive law :

$$\begin{aligned}
&\sum_{\sim\{X\}} g(X, X_1, \ldots, X_{n-1}) \\
&\quad = \sum_{\boldsymbol{V}_1} \sum_{\boldsymbol{V}_2} \cdots \sum_{\boldsymbol{V}_K} F_1(X, \boldsymbol{V}_1)\, F_2(X, \boldsymbol{V}_2) \cdots F_K(X, \boldsymbol{V}_K) \\
&\quad = \left(\sum_{\boldsymbol{V}_1} F_1(X, \boldsymbol{V}_1) \right) \left(\sum_{\boldsymbol{V}_2} F_2(X, \boldsymbol{V}_2) \right) \cdots \left(\sum_{\boldsymbol{V}_K} F_K(X, \boldsymbol{V}_K) \right) \\
&\quad = \prod_{k=1}^{K} \sum_{\sim\{X\}} F_k(X, \boldsymbol{V}_k); \qquad (1.9)
\end{aligned}$$

this shows that the marginal relative to X of the function g is the product of the marginals relatives to X of the functions F_k.

We now focus on the marginal of F_k, for example when $k = 1$. As shown in fig. 1.4, the function $F_1(X, \boldsymbol{V}_1)$ is the product of all functions in the subtree rooted in f_1 of the tree T :

$$F_1(X, \boldsymbol{V}_1) = f_1(X, X_1, \ldots, X_L) G_1(X_1, \boldsymbol{V}_{11}) G_2(X_2, \boldsymbol{V}_{12}) \cdots G_L(X_L, \boldsymbol{V}_{1L}), \quad (1.10)$$

with arguments of g re-numbered so that $f_1(X, X_1, \ldots, X_L)$ is the first neighbor of X, and where G_l is the product of all functions descendant of X_l, $l = 1, \ldots, L$. Note that $\{X_1, \ldots, X_L\}, \boldsymbol{V}_{11}, \ldots, \boldsymbol{V}_{1L}$ is a partition of $\boldsymbol{V}_1$, and using the distributive law again yields

$$\begin{aligned}
&\sum_{\sim\{X\}} F_1(X, \boldsymbol{V}_1) \\
&\quad = \sum_{\sim\{X\}} f_1(X, X_1, \ldots, X_L) G_1(X_1, \boldsymbol{V}_{11}) \cdots G_L(X_L, \boldsymbol{V}_{1L}) \\
&\quad = \sum_{X_1, \ldots, X_L} f_1(X, X_1, \ldots, X_L) \left(\sum_{\boldsymbol{V}_{11}} G_1(X_1, \boldsymbol{V}_{11}) \right) \cdots \left(\sum_{\boldsymbol{V}_{1L}} G_L(X_L, \boldsymbol{V}_{1L}) \right) \\
&\quad = \sum_{\sim\{X\}} \left(f_1(X, X_1, \ldots, X_L) \prod_{l=1}^{L} \sum_{\sim\{X_l\}} G_l(X_l, \boldsymbol{V}_{1l}) \right). \qquad (1.11)
\end{aligned}$$

Thus, if $f_1(X, X_1, \ldots, X_L)$ is a neighbor of X, calculating the marginal of $F_1(X, \boldsymbol{V}_1)$ with respect to X is done as follows :

1. For each neighbor X_l of f_1 other than X, calculate the marginal with respect to X_l of the product of functions in the subtree rooted in X_l.
2. Make the product of these marginals with f_1, then take the marginal with respect to X.

This approach is applied recursively, for each calculation of a marginal function. This generalizes the substitutions presented in fig. 1.3.

1.2.2 Calculation of a marginal

In the following we address the calculation of marginal functions directly on the factor graph, instead of using expression trees. In this context, the calculation of a marginal can be seen as an algorithm operating on the graph; the algorithm consists in making messages (i.e., functions) flow on the edges of the graph and processing them at the nodes.

The algorithm that calculates the marginal $g_i(X_i)$ operates on the acyclic factor graph rooted in X_i. The algorithm starts at the leaves of the tree : each variable node sends a trivial identity function to its parent. Next, each node of the graph waits until it receives messages from all its descendants, then applies the appropriate operation and sends the resulting message to its parent. As shown in fig. 1.3, a variable node simply sends to its parent the product of its incoming messages, whereas a factor node f of parent X makes the product of its incoming messages with f, takes the marginal with respect to X and sends the message to the node X. The algorithm ends at the root X_i, where the marginal $g_i(X_i)$ is the product of all messages received at the node X_i.

Note that messages passing on the edges incident on a variable node X are functions of the variable X only.

1.2.3 Calculation of all marginals : the sum-product algorithm

The preceding developments become attractive when it comes to calculating the marginals $g_i(X_i)$ for several values of i. Applying the algorithm of the preceding section, separately for each i, is not efficient since many intermediary calculations are identical for different values of i. The efficient way of calculating all marginals at once is to overlay, on a single graph, one instance of the algorithm for each i. The resulting algorithm is called the *sum-product algorithm*. This algorithm no longer considers a particular

node as a root, and consequently the parent-child relations in the graph are no longer defined. Instead each node W neighbor of V will be considered at some step as the parent of V. The computation of the message passing from V to W is as in the algorithm for a single marginal, when W is considered as the parent of V and all other neighbors of V as its children.

The sum-product algorithm starts at the leaves of the tree, as does the algorithm for a single marginal. Each other node V stays inactive until it has received messages from all its neighbors but one. A that moment, as in the algorithm for a single marginal, the node V calculates and sends the message for its remaining neighbor, say W, considered as its parent. The node V is then again inactive until it receives a message back from W. Upon reception of that message, the node V calculates and sends the messages for all its neighbors other than W, each being in turn considered as the parent. The algorithm ends when two messages have passed over each edge, one in each direction. For all i, the marginal $g_i(X_i)$ is obtained by making the product of all messages received at the node X_i.

Summary Before giving an example, we summarize the results presented so far. Let $E = \{X, f\}$ be an edge of a factor graph connecting the variable node X with the factor node f. The message passing from the node X to the node f is written $\mu_{X\to f}(X)$, and the message in the opposite direction $\mu_{f\to X}(X)$. Let also $\boldsymbol{N}(V)$ be the set of neighbors of a node V. The propagation and processing rules are as follows :

From a variable node to a factor node :

$$\mu_{X\to f}(X) = \prod_{h\in \boldsymbol{N}(X)\setminus\{f\}} \mu_{h\to X}(X). \tag{1.12}$$

From a factor node to a variable node :

$$\mu_{f\to X}(X) = \sum_{\sim\{X\}} \left(f(\boldsymbol{X}) \prod_{Y\in \boldsymbol{N}(f)\setminus\{X\}} \mu_{Y\to f}(Y) \right), \tag{1.13}$$

where $\boldsymbol{X} = \boldsymbol{N}(f)$ is the set of arguments of the function f.

Example 1.2 : Sum-product algorithm

Considering again the function of example 1.1, the sum-product algorithm follows the

steps indicated in fig. 1.5. Each arrow represents a message, and the circled numbers

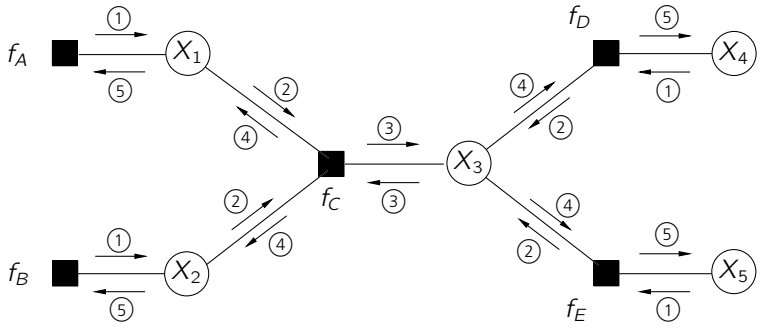


Figure 1.5: Progress of the sum-product algorithm.

indicate at which step the message passes.

The algorithm progresses as follows :

Step ① :

$$\begin{aligned}
\mu_{f_A \to X_1}(X_1) &= \sum_{\sim\{X_1\}} f_A(X_1) = f_A(X_1) \\
\mu_{f_B \to X_2}(X_2) &= \sum_{\sim\{X_2\}} f_B(X_2) = f_B(X_2) \\
\mu_{X_4 \to f_D}(X_4) &= 1 \\
\mu_{X_5 \to f_E}(X_5) &= 1
\end{aligned}$$

Step ② :

$$\begin{aligned}
\mu_{X_1 \to f_C}(X_1) &= \mu_{f_A \to X_1}(X_1) \\
\mu_{X_2 \to f_C}(X_2) &= \mu_{f_B \to X_2}(X_2) \\
\mu_{f_D \to X_3}(X_3) &= \sum_{\sim\{X_3\}} \mu_{X_4 \to f_D}(X_4)\, f_D(X_3, X_4) \\
\mu_{f_E \to X_3}(X_3) &= \sum_{\sim\{X_3\}} \mu_{X_5 \to f_E}(X_5)\, f_E(X_3, X_5)
\end{aligned}$$

Step ③ :

$$\begin{aligned}
\mu_{f_C \to X_3}(X_3) &= \sum_{\sim\{X_3\}} \mu_{X_1 \to f_C}(X_1)\, \mu_{X_2 \to f_C}(X_2)\, f_C(X_1, X_2, X_3) \\
\mu_{X_3 \to f_C}(X_3) &= \mu_{f_D \to X_3}(X_3)\, \mu_{f_E \to X_3}(X_3)
\end{aligned}$$

Step ④ :

$$\begin{aligned}
\mu_{f_C \to X_1}(X_1) &= \sum_{\sim\{X_1\}} \mu_{X_3 \to f_C}(X_3)\, \mu_{X_2 \to f_C}(X_2)\, f_C(X_1, X_2, X_3) \\
\mu_{f_C \to X_2}(X_2) &= \sum_{\sim\{X_2\}} \mu_{X_3 \to f_C}(X_3)\, \mu_{X_1 \to f_C}(X_1)\, f_C(X_1, X_2, X_3) \\
\mu_{X_3 \to f_D}(X_3) &= \mu_{f_C \to X_3}(X_3)\, \mu_{f_E \to X_3}(X_3) \\
\mu_{X_3 \to f_E}(X_3) &= \mu_{f_C \to X_3}(X_3)\, \mu_{f_D \to X_3}(X_3)
\end{aligned}$$

Step ⑤ :

$$\begin{aligned}
\mu_{X_1 \to f_A}(X_1) &= \mu_{f_C \to X_1}(X_1) \\
\mu_{X_2 \to f_B}(X_2) &= \mu_{f_C \to X_2}(X_2) \\
\mu_{f_D \to X_4}(X_4) &= \sum_{\sim\{X_4\}} \mu_{X_3 \to f_D}(X_3)\, f_D(X_3, X_4) \\
\mu_{f_E \to X_5}(X_3) &= \sum_{\sim\{X_5\}} \mu_{X_3 \to f_E}(X_3)\, f_E(X_3, X_5)
\end{aligned}$$

Termination : When all messages have been calculated, the marginal with respect to a variable X_i is the product of all messages incoming into the variable node X_i, thus

$$\begin{aligned}
g_1(X_1) &= \mu_{f_A \to X_1}(X_1)\, \mu_{f_C \to X_1}(X_1) \\
g_2(X_2) &= \mu_{f_B \to X_2}(X_2)\, \mu_{f_C \to X_2}(X_2) \\
g_3(X_3) &= \mu_{f_C \to X_3}(X_3)\, \mu_{f_D \to X_3}(X_3)\, \mu_{f_E \to X_3}(X_3) \\
g_4(X_4) &= \mu_{f_D \to X_4}(X_4) \\
g_5(X_5) &= \mu_{f_E \to X_5}(X_5).
\end{aligned}$$

■

1.3 CYCLIC GRAPHS

As seen in the preceding section, the sum-product algorithm applies in principle only to acyclic factor graphs. In practice however, several applications use it successfully on cyclic graphs, e.g. for decoding turbo-codes [15, 16] or low-density parity check (LDPC)

codes [17].

This section first shows that the presence of cycles in a graph makes the sum-product algorithm iterative, and then briefly discusses the sub-optimality due to the presence of cycles.

1.3.1 Cycles induce iterations

Applying the message update rules to a cyclic graph induces endless updates of the messages : the sum-product algorithm becomes iterative. Consider for example a node V in a cycle. A message incoming in V triggers an update of the messages leaving V on all other edges. Such updates propagate along the cycle so that at some point a new message arrives in V, replacing the previous one, and again triggers an update of all messages leaving V — and so on. These endless updates make the sum-product algorithm iterative.

The initialization of the algorithm must be adapted. Since a message leaving a node V depends on messages received on all other edges incident to V, the initialization of nodes in cycles is a chicken-and-egg problem. To get round this problem, all messages can be initialized to identity messages (constant functions); another option is to choose an initialization depending on the application.

Since the algorithm is iterative, the scheduling of messages updates is no longer unique. The choice of a scheduling is arbitrary, although all schedulings are not necessarily equivalent. All schedulings are theoretical infinite since updates propagate endlessly; however, in many practical applications, messages converge with iterations so that the outputs of the algorithm no longer change, up to machine precision, after a sufficient (finite) number of iterations.

When the algorithm terminates, the approximation of the marginal relative to X_i is obtained as in the acyclic case by calculating the product of all messages received at the variable node X_i.

1.3.2 Sub-optimality due to cycles

The sum-product algorithm applied to a cyclic graph does no longer compute the marginals of the considered function, and the exact link between what the algorithm computes and the true marginals is in general difficult to establish. In spite of that, the sum-product algorithm has been successfully used in several applications with cyclic

graphs, suggesting that the outputs of the algorithm can in some cases be a sufficient approximation of the true marginals.

The fact that the algorithm becomes iterative when the graph contains cycles implies that some pieces of information are taken into account several times along iterations. While this biases the calculation of the marginals, in some applications the cycles are long enough to keep this bias seemingly negligible. Also, in some particular cases, the outputs of the algorithm can be related to the exact marginals (see e.g. [18, 19]). For example if all factors are gaussian, the algorithm converges to marginals with exact averages, but wrong variances.

1.4 SUB-OPTIMAL VARIANT OF THE SUM-PRODUCT ALGORITHM

Previous sections have defined the sum-product algorithm for a function g taking its values in a semiring. The operations '$+$' et '$\times$' are defined accordingly; in particular they satisfy the distributive law. Here we have implicitly considered the semiring consisting of the set of real numbers $\mathbb{R}$ with the usual operations '$+$' (addition) and '$\times$' (multiplication).

As an approximation, each sum could be replaced by its dominant term; in other words, the usual addition '$+$' could be replaced by the operator max. The set $\mathbb{R}$ equipped with the operations 'max' et '$\times$' is also a semiring, and in particular the distributive law is satisfied :

$$X(\max(Y, Z)) = \max(XY, XZ). \tag{1.14}$$

In this semi-ring, the sum-product algorithm is called the max-product algorithm. It can be used as a sub-optimal variant of the sum-product algorithm, to calculate approximations of the marginals of a function.

When working in the logarithmic domain, the product operation becomes a sum. Again, the set $\mathbb{R}$ equipped with the operations 'max' et '$+$' is a semiring, and the distributive law holds :

$$X + \max(Y, Z) = \max(X + Y, X + Z). \tag{1.15}$$

The max-product algorithm in this semi-ring is called the max-sum algorithm. It is equivalent to the max-product algorithm, only that its outputs are expressed in the log-domain.

1.5 APPLICATIONS

The sum-product algorithm can be applied to virtually any problem of function marginalization, provided that the functions have a suitable factorization. It has applications, for example, in artificial intelligence, in telecommunications or in signal processing. Particular cases are the belief propagation algorithm due to Pearl [14, 20], an FFT computation algorithm [10], the so-called BCJR algorithm for decoding convolutional codes (see section 2.2), the turbo-decoding algorithm (see section 2.3), etc.

To summarize, the general procedure for applying the factor graph framework to a marginalization problem is as follows :

1. Find a suitable factorization of the considered function.
2. Draw the corresponding factor graph.
3. Apply the sum-product algorithm to this graph.

Chapter 2 uses this framework to derive several decoding algorithms. It is then applied in chapter 5 to design receivers for communication over the relay channel.

Codes and Factor graphs

2

The factor graphs framework offers a unified way of deriving decoding algorithms; it has been successfully applied, for example, to important families of codes such as turbo-codes or low-density parity check codes (LDPC codes) [13, 21]. This chapter surveys several decoding algorithms that are used later in this thesis as part of larger receivers.

The chapter first addresses convolutional codes, and derives the so-called BCJR decoding algorithm in the factor graphs framework. Turbo-codes are addressed next, and provide a first example of cyclic factor graph resulting in an iterative decoding algorithm. Then comes the decoder of bit-interleaved coded modulation (BICM), also addressed in prevision of future chapters.

The chapter ends with a presentation of two tools for analyzing iterative receivers. The first analysis tool is the extrinsic information transfer chart (EXIT chart), and the second one is an analytical prediction tool for the convergence threshold of turbo-codes.

This analytical prediction tool is an original contribution of this thesis. It was published in [1, 2].

Contents

2.1 RECEIVER DESIGN IN THE FACTOR GRAPHS FRAMEWORK

This sections formulates the design of optimal receivers in the maximum a posteriori (MAP) sense, using the factor graphs framework [22]. Using the factor graphs framework is relevant because MAP decoding amounts to marginalizing some objective function, as shown below.

A sequence of independent equiprobable information bits $\boldsymbol{U}$ of length K is encoded into a binary codeword $\boldsymbol{X}$, modulated into a sequence of symbols $\boldsymbol{S}$ of length N. These symbols are sent over a communication channel, and received as a sequence $\boldsymbol{Y}$ (fig. 2.1). The channel is characterized by its transition probabilities $P(\boldsymbol{Y}|\boldsymbol{S})$.

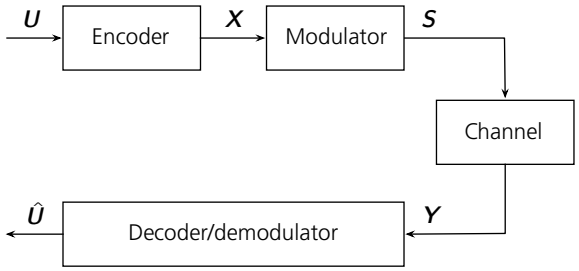


Figure 2.1: Transmission scheme.

After the receiver observes $\boldsymbol{Y} = \boldsymbol{y}$, decoding the sequence $\boldsymbol{U}$ in the MAP sense consists in finding the values $\hat{U}_k$ maximizing the probability mass function $P(U_k|\boldsymbol{Y} = \boldsymbol{y})$, for each information bit $k = 1, \ldots, K$. This problem can be recast as a marginalization problem, since

$$\hat{U}_k = \arg \max_{U_k=-1,1} P(U_k|\boldsymbol{Y} = \boldsymbol{y}) \tag{2.1}$$

$$= \arg \max_{U_k=-1,1} \sum_{\sim\{U_k\}} P(\boldsymbol{U}|\boldsymbol{Y} = \boldsymbol{y}). \tag{2.2}$$

Using the Bayes rule and the fact that the maximization problem is independent of $P(\boldsymbol{Y} = \boldsymbol{y})$, we can write

$$\hat{U}_k = \arg \max_{U_k=-1,1} \sum_{\sim\{U_k\}} P(\boldsymbol{y}|\boldsymbol{U}) \cdot P(\boldsymbol{U}). \tag{2.3}$$

Since the information bits $\boldsymbol{U}$ are independent and equiprobable, the factor $P(\boldsymbol{U})$ is

constant and can be omitted. Making $\boldsymbol{X}$ and $\boldsymbol{S}$ apparent in the expression above gives

$$\hat{U}_k = \arg \max_{U_k=-1,1} \sum_{\sim\{U_k\}} \sum_{\boldsymbol{X},\boldsymbol{S}} P(\boldsymbol{y}|\boldsymbol{S},\boldsymbol{X},\boldsymbol{U}) \cdot P(\boldsymbol{S}|\boldsymbol{X},\boldsymbol{U}) \cdot P(\boldsymbol{X}|\boldsymbol{U}). \tag{2.4}$$

The summations over $\boldsymbol{X}$ and $\boldsymbol{S}$ can be omitted since the notation $\sum_{\sim\{U_k\}}$ already indicates that *all* variables are being summed over, except U_k. Using that $\boldsymbol{U}-\boldsymbol{X}-\boldsymbol{S}-\boldsymbol{Y}$ is a Markov chain yields

$$P(\boldsymbol{y}|\boldsymbol{S},\boldsymbol{X},\boldsymbol{U}) = P(\boldsymbol{y}|\boldsymbol{S}) \quad \text{and} \tag{2.5}$$

$$P(\boldsymbol{S}|\boldsymbol{X},\boldsymbol{U}) = P(\boldsymbol{S}|\boldsymbol{X}). \tag{2.6}$$

We finally obtain, from (2.4),

$$\hat{U}_k = \arg \max_{U_k=-1,1} \sum_{\sim\{U_k\}} P(\boldsymbol{y}|\boldsymbol{S}) \cdot P(\boldsymbol{S}|\boldsymbol{X}) \cdot P(\boldsymbol{X}|\boldsymbol{U}). \tag{2.7}$$

Estimating an information bit U_k in the MAP sense requires thus to marginalize an objective function $F(\boldsymbol{U},\boldsymbol{X},\boldsymbol{S})$ defined by

$$F(\boldsymbol{U},\boldsymbol{X},\boldsymbol{S}) \triangleq P(\boldsymbol{y}|\boldsymbol{S}) \cdot P(\boldsymbol{S}|\boldsymbol{X}) \cdot P(\boldsymbol{X}|\boldsymbol{U}). \tag{2.8}$$

Note that $\boldsymbol{y}$ is an observation and not a variable; it is thus not considered as an argument of the objective function.

The first factor of the objective function, $P(\boldsymbol{y}|\boldsymbol{S})$, accounts for the effect of the communication channel. The factors $P(\boldsymbol{S}|\boldsymbol{X})$ and $P(\boldsymbol{X}|\boldsymbol{U})$ account for modulation an coding; they characterize the unique correspondence, respectively, between symbols $\boldsymbol{S}$ and codewords $\boldsymbol{X}$, and between codewords $\boldsymbol{X}$ and information bits $\boldsymbol{U}$. They are equal to 1 if and only if the correspondence is valid, and to 0 otherwise. We write this

$$P(\boldsymbol{S}|\boldsymbol{X}) = \mathbb{I}\{\boldsymbol{S} = \text{out}_M(\boldsymbol{X})\} \tag{2.9}$$

$$P(\boldsymbol{X}|\boldsymbol{U}) = \mathbb{I}\{\boldsymbol{X} = \text{out}_C(\boldsymbol{U})\} \tag{2.10}$$

where $\text{out}_C()$ is the encoding function, $\text{out}_M()$ the modulation function, and where $\mathbb{I}()$ is an indicator function equal to 1 if its boolean argument is true, and equal to 0 otherwise.

All these factors of the objective function $F(\boldsymbol{U},\boldsymbol{X},\boldsymbol{S})$ can usually be factorized further by taking into account the channel model, the type of modulation and the type of

coding. Using the resulting factorization, the sum-product algorithm can calculate the marginals and finally find estimates of the bits $\boldsymbol{U}$ as in (2.7).

The following sections detail these steps for several families of codes.

2.2 CONVOLUTIONAL CODES

This section first defines convolutional codes, and then presents the derivation of the appropriate bitwise MAP decoding algorithm.

2.2.1 Definition and encoding

A convolutional code transforms a continuous stream of information bits into a continuous stream of coded bits [23]. The encoding operation consists in shifting the sequence into a shift register, from the contents of which coded bits are calculated. The redundancy introduced by this operation gives to the sequence some resilience against channel degradations.

The encoding operation is commonly represented by the generator matrix $G(D)$ of size $r \times n$ ($n > r$) and rank r; elements of this matrix are polynomials in the delay operator D with binary coefficients :

$$G(D) = \begin{pmatrix} g_{1,1}(D) & g_{1,2}(D) & \cdots & g_{1,n}(D) \\ g_{2,1}(D) & g_{2,2}(D) & \cdots & g_{2,n}(D) \\ \vdots & \vdots & \ddots & \vdots \\ g_{r,1}(D) & g_{r,2}(D) & \cdots & g_{r,n}(D) \end{pmatrix}. \tag{2.11}$$

The encoder represented by this matrix associates r information sequences $U_1(D), \ldots, U_r(D)$ to n coded sequences $X_1(D), \ldots, X_n(D)$ as follows :

$$(X_1(D) \ \ldots \ X_n(D)) = (U_1(D) \ \ldots \ U_r(D)) \ G(D). \tag{2.12}$$

Such an encoder, associating r information sequences to n coded sequences is said to have a code rate $R = r/n$. To simplify notations we restrict ourselves to the case $r = 1$, the extension to the general case being straightforward. We omit the corresponding indices.

The encoding operation is as follows. The encoder produces, for the k^{th} information bits U_k, n coded bits $X_k^1, \ldots, X_k^n$. In particular, the l^{th} coded bit X_k^l (output corre-

sponding to the polynomial $g_l(D)$) is obtained by

$$X_k^l = \sum_{m=0}^{M} g_l^{(m)} U_{k-m} \mod 2 \tag{2.13}$$

where $g_l^{(m)}$ is the m^{th} coefficient of the polynomial $g_l(D)$. The bits produced by the n polynomials are then concatenated into a single sequence

$$(\ldots, X_k^1, \ldots, X_k^n, X_{k+1}^1, \ldots, X_{k+1}^n, \ldots). \tag{2.14}$$

The largest degree M of the polynomials in $G(D)$ is called the memory of the encoder, and the parameter $K_c = M + 1$ is its constraint length.

If any of the polynomials is 1, the information sequence is included unmodified in the coded sequence. The code is then said *systematic*. Otherwise it is said *non-systematic*; such codes are usually refered to as *Non-Systematic Convolutional* codes (NSC codes).

Example 2.1 : NSC encoder

The encoder with generator matrix

$$G(D) = \begin{pmatrix} 1 + D + D^2 & 1 + D^2 \end{pmatrix}$$

has a memory $M = 2$ and a code rate $R = \frac{1}{2}$. This matrix can be represented by the octal representation of the binary coefficients of the polynomials, here $(5_8, 7_8)$. The corresponding code is a NSC code.

The implementation of this encoder with a shift register is represented in fig. 2.2. ■

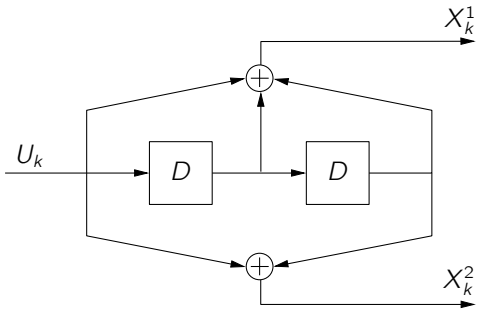


Figure 2.2: NSC encoder with polynomials $(5_8, 7_8)$ (octal notation).

Another family of convolutional codes have encoders with a feedback loop from the contents of the register to its input. Such codes are said *recursive*. In particular, recursive systematic convolutional codes (RSC codes) have a generator matrix

$$G(D) = \begin{pmatrix} 1 & \frac{g_2(D)}{g_1(D)} & \cdots & \frac{g_n(D)}{g_1(D)} \end{pmatrix} \tag{2.15}$$

When the bit U_k enters the encoder, the feedback loop produces the following input in the register

$$A_k = U_k + \sum_{m=1}^{M} g_1^{(m)} A_{k-m} \mod 2, \tag{2.16}$$

where the $A_{k-1}, \ldots, A_{k-M}$ are the contents of the register. The non-systematic coded bits are calculated from the contents of the register as follows :

$$X_k^l = \sum_{m=0}^{M} g_l^{(m)} A_{k-m} \mod 2, \tag{2.17}$$

whereas the systematic output is of course $X_k^1 = U_k$.

Example 2.2 : RSC encoder

The RSC encoder with generator matrix

$$G(D) = \begin{pmatrix} 1 & \frac{1+D^2}{1+D+D^2} \end{pmatrix}$$

is depicted in fig. 2.3. ■

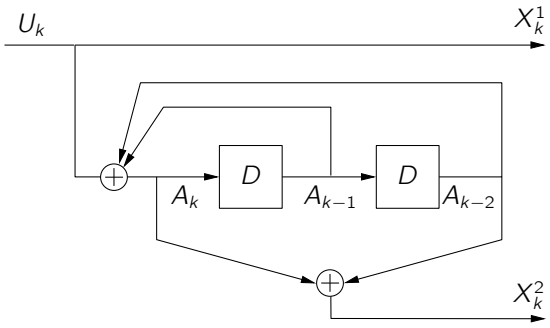


Figure 2.3: RSC encoder with generator matrix $G(D) = \begin{pmatrix} 1 & \frac{1+D^2}{1+D+D^2} \end{pmatrix}$.

2.2.2 Optimal MAP decoding : the BCJR algorithm

The optimal bitwise MAP decoding of convolutional codes maximizes the a posteriori probabilities of the information bits. The usual algorithm to achieve this is the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [24].

This algorithm is actually a particular instance of the sum-product algorithm, applied to the factor graph of the a posteriori probability mass function of the information sequence.

Factorization of the objective function

The first step for deriving the BCJR algorithm in the factor graphs framework is to factorize the a posteriori probability mass function of the information sequence, as initiated in section 2.1. We continue the factorization here, taking into account the channel model and the encoding method. Since the focus is on decoding the code, we simply use binary-phase shift keying (BPSK) modulation.

As discussed above, optimal bitwise decoding in the MAP sense amounts to marginalizing an objective function $F(\boldsymbol{U}, \boldsymbol{X}, \boldsymbol{S})$:

$$F(\boldsymbol{U}, \boldsymbol{X}, \boldsymbol{S}) \triangleq P(\boldsymbol{y}|\boldsymbol{S}) \cdot P(\boldsymbol{S}|\boldsymbol{X}) \cdot P(\boldsymbol{X}|\boldsymbol{U}). \qquad (2.8)$$

This function must be factorized further in order to make the sum-product algorithm more efficient.

Factorization of $P(\boldsymbol{y}|\boldsymbol{S}) \cdot P(\boldsymbol{S}|\boldsymbol{X})$ The first factor depends on the channel model and the second one on the modulation. For a memoryless channel and BPSK modulation, we simply have

$$P(\boldsymbol{y}|\boldsymbol{S}) \cdot P(\boldsymbol{S}|\boldsymbol{X}) = \prod_{k=1}^{K} \prod_{l=1}^{n} P(y_k^l|S_k^l) \cdot \mathbb{I}\{S_k^l = \mathrm{out}_M(X_k^l)\} \qquad (2.18)$$

where y_k^l is the channel observation for the transmitted symbol S_k^l, which corresponds to the coded bit X_k^l. We can make the correspondence between BPSK symbols S_k^l and coded bits X_k^l implicit by writing

$$P(\boldsymbol{y}|\boldsymbol{S}) \cdot P(\boldsymbol{S}|\boldsymbol{X}) = \prod_{k=1}^{K} \prod_{l=1}^{n} P(y_k^l|X_k^l). \qquad (2.19)$$

Factorization of $P(\boldsymbol{X}|\boldsymbol{U})$ The second factor,

$$P(\boldsymbol{X}|\boldsymbol{U}) = \mathbb{I}\{\boldsymbol{X} = \text{out}_C(\boldsymbol{U})\}, \qquad (2.10)$$

factorizes easily by noting that the output of the convolutional encoder is a Markov process : the outputs of the encoder depend only on the current input U_k and on the state $\boldsymbol{T_k}$ of the encoder memory. For a non-recursive convolutional encoder, the state of the memory is $\boldsymbol{T_k} = [U_{k-1}, \ldots, U_{k-M}]$, whereas for a recursive encoder it is $\boldsymbol{T_k} = [A_{k-1}, \ldots, A_{k-M}]$. Consequently we can write

$$\begin{aligned} \mathbb{I}\{\boldsymbol{X} = \text{out}_C(\boldsymbol{U})\} = \mathbb{I}\{\boldsymbol{T_1} = \mathbf{0}\} \quad \prod_k \quad & \mathbb{I}\{\boldsymbol{T_{k+1}} \leftarrow (U_k, \boldsymbol{T_k})\} \\ \times \quad & \mathbb{I}\{X_k^1, \ldots, X_k^n = \text{out}_C(U_k, \boldsymbol{T_k})\}. \end{aligned} \qquad (2.20)$$

The meaning of this expression is the following : $\boldsymbol{X}$ is the codeword corresponding to $\boldsymbol{U}$ if, for all k, the following conditions are fulfilled :

- The outputs of the encoder $X_k^1, \ldots, X_k^n$ at time k correspond to its current state $\boldsymbol{T_k}$ and to the input bit U_k;
- The current state $\boldsymbol{T_k}$ and the input bit U_k indeed produce a transition to the state $\boldsymbol{T_{k+1}}$.

Moreover, requiring the memory of the encoder to be initialized with zeros yields the factor $\mathbb{I}\{\boldsymbol{T_1} = \mathbf{0}\}$. When the encoder encodes finite blocks of data instead of a continuous stream, one can also make the encoder terminate the blocks in the state zero by adding some bits at the tail of each codeword; in that case the expression (2.20) is multiplied by an additional factor $\mathbb{I}\{\boldsymbol{T_{K+1}} = \mathbf{0}\}$.

Factor graph

The factor graph is made as usual by creating nodes for each factor and each variable, and by connecting factors to variables the depend on. Its general structure is presented in fig. 2.4, and its k^{th} section is detailed in fig. 2.5.

All other sections of the graph are identical, behalves the section corresponding to U_1 which has an additional factor node $\mathbb{I}\{\boldsymbol{T_1} = \mathbf{0}\}$ in accordance with (2.20), and possibly the last section which has a similar factor node when the encoder is required to finish blocks in the zero state (in the case of block-by-block encoding).

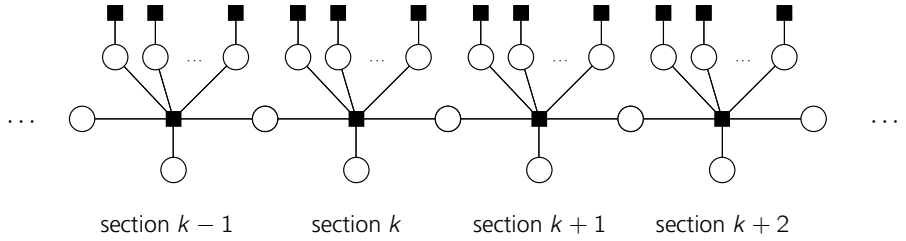


Figure 2.4: Structure of the factor graph for the derivation of the BCJR algorithm.

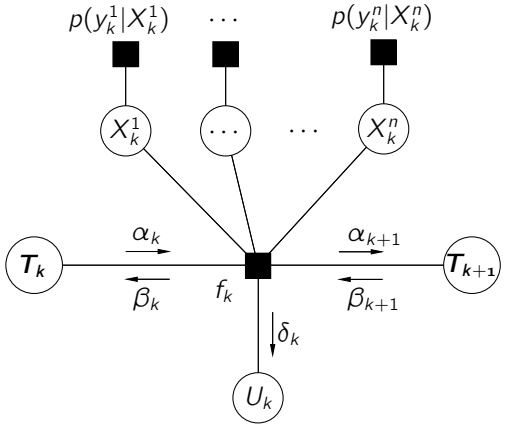


Figure 2.5: Detail of the k^{th} section of the factor graph used for the derivation of the BCJR algorithm. The factor node f_k represents $\mathbb{I}\{\boldsymbol{T_{k+1}} \leftarrow (U_k, \boldsymbol{T_k})\}\mathbb{I}\{X_k^1, \ldots, X_k^n = \text{out}_C(U_k, \boldsymbol{T_k})\}$.

Application of the sum-product algorithm

Since the graph is a tree, the sum-product algorithm produces exact marginals. The marginal with respect to a variable U_k is the product of all incoming messages at node U_k : here the only message is $\delta_k(U_k)$ (fig. 2.5). In the following we detail the algorithm for obtaining $\delta_k(U_k)$.

Initialization Initialization happens at the leaves of the graph.

- The nodes $P(y_k^1|X_k^1), \ldots, P(y_k^n|X_k^n)$ send messages equal to their local function to their neighbors, respectively $X_k^1, \ldots, X_k^n$. Since the nodes X have degree 2, they simply forward the message to the node f_k.
- At the variable node U_k, a trivial identity message (constant function) is sent to f_k; this has no effect.

- The factor node $\mathbb{I}\{\boldsymbol{T_1} = \mathbf{0}\}$ sends a message α_0 equal to itself, to its neighbor node $\boldsymbol{T_1}$.
- If it exists, the factor node $\mathbb{I}\{\boldsymbol{T_{K+1}} = \mathbf{0}\}$ sends a message β_{K+1} equal to itself to its neighbor $\boldsymbol{T_{K+1}}$. If this factor node does not exist, then the variable node $\boldsymbol{T_{K+1}}$ is the end of the graph; in that case the message β_{K+1} is initialized as an identity message.

Propagation At the node f_k, the propagation and processing rules of the sum-product algorithm give the following recursions :

- Message sent to $\boldsymbol{T_{k+1}}$:

$$\alpha_{k+1}(\boldsymbol{T_{k+1}}) = \sum_{\sim\{\boldsymbol{T_{k+1}}\}} \alpha_k(\boldsymbol{T_k}) \cdot \mathbb{I}\{\boldsymbol{T_{k+1}} \leftarrow (U_k, \boldsymbol{T_k})\} \times \mathbb{I}\{X_k^1, \ldots, X_k^n = \mathrm{out}_C(U_k, \boldsymbol{T_k})\} \prod_{l=1}^{n} P(y_k^l | X_k^l). \quad (2.21)$$

- Message sent to $\boldsymbol{T_k}$:

$$\beta_k(\boldsymbol{T_k}) = \sum_{\sim\{\boldsymbol{T_k}\}} \beta_{k+1}(\boldsymbol{T_{k+1}}) \cdot \mathbb{I}\{\boldsymbol{T_{k+1}} \leftarrow (U_k, \boldsymbol{T_k})\} \times \mathbb{I}\{X_k^1, \ldots, X_k^n = \mathrm{out}_C(U_k, \boldsymbol{T_k})\} \prod_{l=1}^{n} P(y_k^l | X_k^l). \quad (2.22)$$

Termination The node f_k sends to U_k the following message :

$$\delta_k(U_k) = \sum_{\sim\{U_k\}} \alpha_k(\boldsymbol{T_k}) \cdot \beta_{k+1}(T_{k+1}) \cdot \mathbb{I}\{\boldsymbol{T_{k+1}} \leftarrow (U_k, \boldsymbol{T_k})\} \times \mathbb{I}\{X_k^1, \ldots, X_k^n = \mathrm{out}_C(U_k, \boldsymbol{T_k})\} \prod_{l=1}^{n} P(y_k^l | X_k^l). \quad (2.23)$$

This message is the marginal relative to U_k of the objective function $F(\boldsymbol{U}, \boldsymbol{X}, \boldsymbol{S})$. Consequently, the bitwise MAP estimate of U_k is the argument that maximizes $\delta_k(U_k)$:

$$\begin{aligned} \hat{U}_k &= \arg \max_{U_k=-1,1} P(U_k|\boldsymbol{y}) && (2.24)\\ &= \arg \max_{U_k=-1,1} \sum_{\sim\{U_k\}} F(\boldsymbol{U}, \boldsymbol{X}, \boldsymbol{S}) && (2.25)\\ &= \arg \max_{U_k=-1,1} \delta_k(U_k). && (2.26) \end{aligned}$$

The expressions above can be simplified further. Because of the indicator functions $\mathbb{I}$, several terms disappear; moreover using the definition

$$\gamma_k(X_k^1, \ldots, X_k^n) \triangleq \prod_{l=1}^{n} P(y_k^l|X_k^l), \tag{2.27}$$

we can write, for a coded sequence $X_k^1, \ldots, X_k^n$ corresponding to an information sequence $U_k, \ldots, U_{k-M}$:

$$\alpha_{k+1}(U_k, \ldots, U_{k-M+1}) = \sum_{U_{k-M}} \alpha_k(U_{k-1}, \ldots, U_{k-M})\, \gamma_k(X_k^1, \ldots, X_k^n), \tag{2.28}$$

$$\beta_k(U_{k-1}, \ldots, U_{k-M}) = \sum_{U_k} \beta_{k+1}(U_k, \ldots, U_{k-M+1})\, \gamma_k(X_k^1, \ldots, X_k^n), \tag{2.29}$$

and

$$\begin{aligned} \delta_k(U_k) = \sum_{U_{k-1},\ldots,U_{k-M}} & \alpha_k(U_{k-1}, \ldots, U_{k-M}) \\ & \times \beta_{k+1}(U_k, \ldots, U_{k-M+1})\, \gamma_k(X_k^1, \ldots, X_k^n). \end{aligned} \tag{2.30}$$

This algorithm, derived as a particular instance of the sum-product algorithm, is actually the BCJR algorithm with its forward recursion α_k, backward recursion β_k and the functions γ_k, written as in the usual formulation of the algorithm.

The messages α_k can be regarded as the probability mass function of the bits $U_{k-1}, \ldots, U_{k-M}$, given the observation of the sequence until time $k-1$:

$$\alpha_k(U_{k-1}, \ldots, U_{k-M}) = P(U_{k-1}, \ldots, U_{k-M}|y_1^1, \ldots, y_{k-1}^n)$$

and conversely, β_{k+1} is the probability mass function of $U_k, \ldots, U_{k-M+1}$ given the observation of the sequence from time $k+1$ until time K :

$$\beta_{k+1}(U_k, \ldots, U_{k-M+1}) = P(U_k, \ldots, U_{k-M+1} | y_{k+1}^1, \ldots, y_K^n).$$

The marginal $\delta_k(U_k)$ with respect to U_k is thus a combination of information over the preceding part of the sequence (through α_k), information over the following part of the sequence (through β_{k+1}), and information over the current observation at time k (through γ_k).

From these marginals relative to U_k, we obtain the log-ratio for each information bit, defined as follows :

$$L_{U_k} \triangleq \ln\left(\frac{\delta_k(U_k = +1)}{\delta_k(U_k = -1)}\right). \tag{2.31}$$

Implementation

To avoid numerical accuracy problems, the BCJR is most often implemented in its logarithmic form (details in appendix 2.A). This logarithmic form of the BCJR algorithm is usually referred to as the LOG-MAP algorithm.

2.2.3 Sub-optimal variant of the BCJR algorithm

A sub-optimal variant of the BCJR algorithm can be derived as above, but in the max-sum semiring. The resulting algorithm is the so-called MAX-LOG-MAP algorithm. Using the following notations :

$$\begin{aligned}
\bar{\alpha}_k(U_{k-1}, \ldots, U_{k-M}) &= \ln(\alpha_k(U_{k-1}, \ldots, U_{k-M})) \\
\bar{\beta}_k(U_{k-1}, \ldots, U_{k-M}) &= \ln(\beta_k(U_{k-1}, \ldots, U_{k-M})) \\
\bar{\gamma}_k(X_k^1, \ldots, X_k^n) &= \ln(\gamma_k(X_k^1, \ldots, X_k^n)),
\end{aligned}$$

the message update rules become, in the max-sum semiring :

$$\bar{\alpha}_{k+1}(U_k, \ldots, U_{k-M+1}) = \max_{U_{k-M}}(\bar{\alpha}_k(U_{k-1}, \ldots, U_{k-M}) + \bar{\gamma}_k(X_k^1, \ldots, X_k^n)) \tag{2.32}$$

$$\bar{\beta}_k(U_{k-1}, \ldots, U_{k-M}) = \max_{U_k}(\bar{\beta}_{k+1}(U_k, \ldots, U_{k-M+1}) + \bar{\gamma}_k(X_k^1, \ldots, Xk^n)) \tag{2.33}$$

and

$$\begin{aligned}\bar{\delta}_k(U_k) = \max_{U_{k-1},\ldots,U_{k-M}} (&\bar{\alpha}_k(U_{k-1},\ldots,U_{k-M}) \\ &+ \bar{\beta}_{k+1}(U_k,\ldots,U_{k-M+1}) + \bar{\gamma}_k(X_k^1,\ldots,X_k^n)).\end{aligned} \tag{2.34}$$

In comparison with the BCJR algorithm, the MAX-LOG-MAP algorithm achieves a slightly lower performance, but with a lower complexity.

2.3 TURBO-CODES

Turbo-codes have initiated a revolution in channel coding, for being the first codes to achieve code rates close to the Shannon capacity on several important channel models [15, 16]. The coding technique consists in combining several simple constituent codes into an efficient global code. MAP decoding of such a code would be far to complex; however very good performance can be achieved by using instead the so-called turbo-decoding algorithm, which is an iterative algorithm based on the decoding algorithms of the constituent codes.

The turbo-decoding algorithm has been shown in [20, 14] to be a particular instance of the sum-product algorithm, applied to the a posteriori probability of the information bits. Since the factor graph of this probability contains cycles, the algorithm is iterative and sub-optimal in comparison with MAP decoding.

The next two sections detail turbo-coding and the turbo-decoding algorithm.

2.3.1 Definition and encoding

This section briefly presents the classical parallel turbo-coding scheme, as initially proposed by Berrou et al. in [15, 16].

Two constituent RSC codes of rate $R = \frac{1}{2}$ are concatenated in parallel as in fig. 2.6. The first code takes the sequence of information bits $\boldsymbol{U}$ as input, while the second works on an interleaved copy of $\boldsymbol{U}$. The constituent codes produce, besides their systematic outputs, parity bits written respectively X_k^1 and X_k^2 ($k = 1,\ldots,K$). The information bits and parity bits are then multiplexed in a single sequence :

$$(\ldots, U_k, X_k^1, X_k^2, U_{k+1}, X_{k+1}^1, X_{k+1}^2, \ldots). \tag{2.35}$$

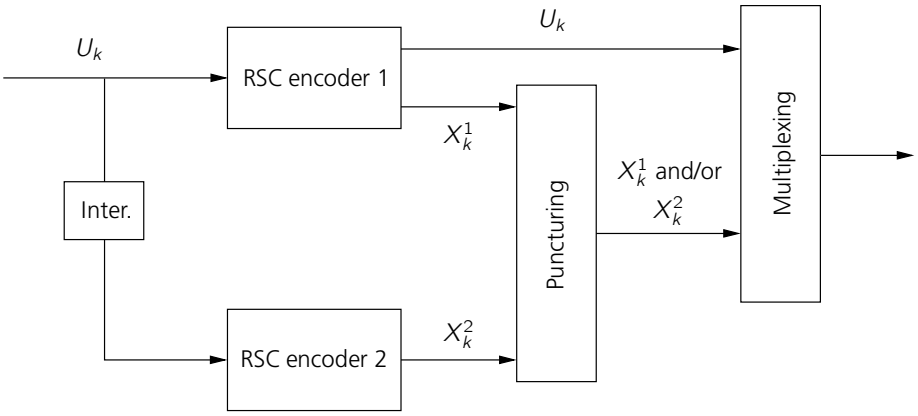


Figure 2.6: Turbo-encoder.

This yields a global code rate $R = \frac{1}{3}$. Higher code rates can be obtained by puncturing, i.e. by throwing some of the parity bits.

Many variants of this construction are possible, by changing the constituent codes, their rates, the puncturing, the interleaver, the number of constituent codes, or the type of concatenation (serial instead of parallel).

2.3.2 Turbo-decoding algorithm

This section derives the turbo-decoding algorithm as a particular instance of the sum-product algorithm.

As seen in section 2.1, MAP decoding consists in finding estimates $\hat{U}_k$ of the information bits as follows :

$$\begin{aligned} \hat{U}_k &= \arg \max_{U_k=-1,1} P(U_k|\boldsymbol{y}) \\ &= \arg \max_{U_k=-1,1} \sum_{\sim\{U_k\}} P(\boldsymbol{y}|\boldsymbol{S}) \cdot P(\boldsymbol{S}|\boldsymbol{X}) \cdot P(\boldsymbol{X}|\boldsymbol{U}). \end{aligned}$$

The marginalization with respect to U_k is carried out in the factor graphs framework, with the objective function $F(\boldsymbol{U}, \boldsymbol{X}, \boldsymbol{S})$:

$$F(\boldsymbol{U}, \boldsymbol{X}, \boldsymbol{S}) \triangleq P(\boldsymbol{y}|\boldsymbol{S}) \cdot P(\boldsymbol{S}|\boldsymbol{X}) \cdot P(\boldsymbol{X}|\boldsymbol{U}) \qquad (2.8)$$

where $\boldsymbol{X}$ is the whole coded sequence, i.e.,

$$(\ldots, U_k, X_k^1, X_k^2, U_{k+1}, X_{k+1}^1, X_{k+1}^2, \ldots) \tag{2.36}$$

and $\boldsymbol{y}$ the corresponding observations at the receiver,

$$(\ldots, y_k^0, y_k^1, y_k^2, y_{k+1}^0, y_{k+1}^1, y_{k+1}^2, \ldots). \tag{2.37}$$

Factorization of the objective function

Factorization of $P(\boldsymbol{y}|\boldsymbol{S}) \cdot P(\boldsymbol{S}|\boldsymbol{X})$ For a memoryless channel and BPSK modulation, we have :

$$P(\boldsymbol{y}|\boldsymbol{S}) \cdot P(\boldsymbol{S}|\boldsymbol{X}) = \prod_k P(y_k^0|U_k) \cdot P(y_k^1|X_k^1) \cdot P(y_k^2|X_k^2), \tag{2.38}$$

where the correspondence between coded bits and BPSK symbols has been made implicit as in (2.19). If some bits have been punctured, the corresponding probabilities in (2.38) are considered as uniform.

Factorization of $P(\boldsymbol{X}|\boldsymbol{U})$ The encoders are convolutional encoders, thus the function factorizes as in (2.20). Writing the interleaved bits $U_{\pi(k)}$, we have

$$\begin{aligned} P(\boldsymbol{X}|\boldsymbol{U}) &= \mathbb{I}\{\boldsymbol{X} = \mathrm{out}_C(\boldsymbol{U})\} \\ &= \mathbb{I}\{\boldsymbol{T_1^1} = \mathbf{0}\} \prod_k \mathbb{I}\{\boldsymbol{T_{k+1}^1} \leftarrow (U_k, \boldsymbol{T_k^1})\} \\ &\qquad \times \mathbb{I}\{X_k^1 = \mathrm{out}_{C,1}(U_k, \boldsymbol{T_k^1})\} \\ &\times \mathbb{I}\{\boldsymbol{T_1^2} = \mathbf{0}\} \prod_k \mathbb{I}\{\boldsymbol{T_{k+1}^2} \leftarrow (U_{\pi(k)}, \boldsymbol{T_k^2})\} \\ &\qquad \times \mathbb{I}\{X_k^2 = \mathrm{out}_{C,2}(U_{\pi(k)}, \boldsymbol{T_k^2})\}. \end{aligned} \tag{2.39}$$

The meaning of this expression is similar to the meaning of (2.20), but here for two encoders. More precisely, it shows that a codeword $\boldsymbol{X}$ corresponds to information bits $\boldsymbol{U}$ if and only if (with $i = 1, 2$) :

- The initial state of each encoder is zero;
- For all k, the state $\boldsymbol{T_{k+1}^i}$ of encoder i is obtained from state $\boldsymbol{T_k^i}$ and input U_k;
- For all k, the output X_k^i results from a state $\boldsymbol{T_k^i}$ and an input U_k.

As for a convolutional code, the final encoder state can be brought to zero.

Factor graph

The factor graph of fig. 2.8 represents the above factorization of the objective function.

We highlight the following :

- Firstly, the graph contains cycles. The sum-product algorithm is thus iterative and does not produce the exact marginals of the objective function; in other words, the outputs of the decoding algorithm are not optimal in the MAP sense.
- Secondly, the factor graph for turbo-decoding includes the factor graphs of the constituent codes. Consequently the turbo-decoding algorithm will include the decoders for the constituent codes, i.e. two BCJR algorithms.
- Finally, since the interleaver connects the graphs of the constituent decoders, messages will be exchanged between these. We detail this below.

Application of the sum-product algorithm

Since the algorithm is iterative, the scheduling of updates can be chosen arbitrarily. Here we follow the usual scheduling, i.e., first decoding the first constituent code, then forwarding its messages to the second decoder through the interleaver, decoding the second code, and carry out several iterations in this order. More precisely, these steps are as follows.

Decoding of a constituent code The decoding algorithm for a constituent code is very similar to the algorithm for this code if it was alone. The BCJR algorithm is only slightly modified to take into account the messages sent by the other constituent decoder.

Using the notations of fig. 2.7 (omitting the index i where no confusion is possible), the information available on U_k before decoding the constituent decoder can be written

$$\mu_k(U_k) = P(y_k^0|U_k) \cdot P_a(U_k); \tag{2.40}$$

it combines information from observing the channel and information from the other constituent decoder. The latter is usually called *a priori information*. At the first iter-

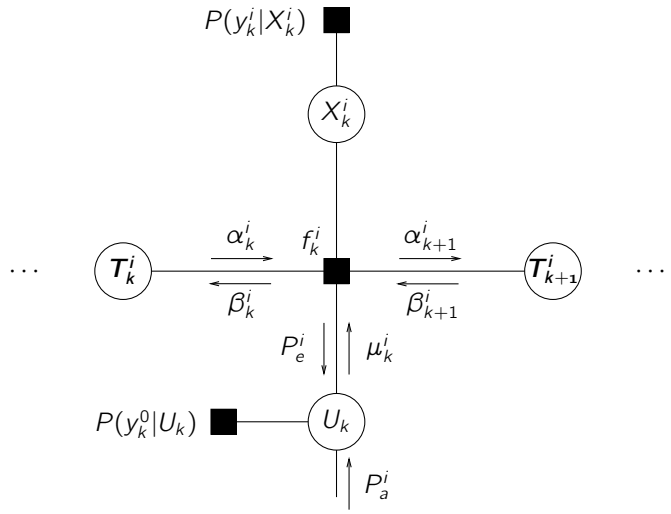


Figure 2.7: The k^{th} section of the constituent decoder $i = 1, 2$ taken from the factor graph of the turbo-decoder. The node f_k^i represents the factor $f_k^i = \mathbb{I}\{\boldsymbol{T_{k+1}^i} \leftarrow (U_k, \boldsymbol{T_k^i})\} \cdot \mathbb{I}\{X_k^i = \text{out}_{C,i}(U_k, \boldsymbol{T_k^i})\}$. The a priori information message P_a^i comes from the other constituent decoder, through the interleaver.

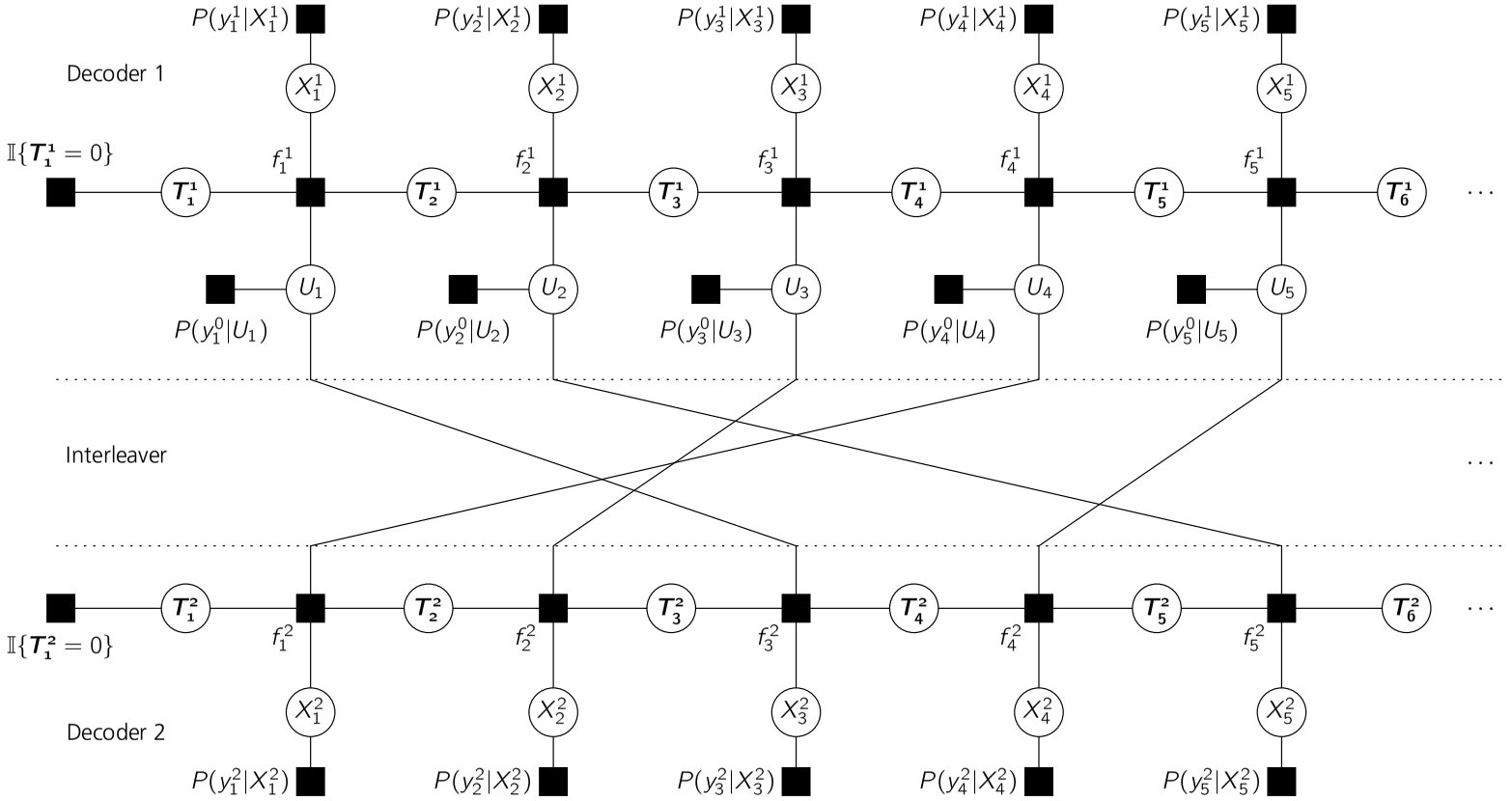


Figure 2.8: Factor graph for turbo-decoding (first sections). The node f_k^i represents the factor $f_k^i = \mathbb{I}\{\boldsymbol{T}_{k+1}^i \leftarrow (U_k, \boldsymbol{T}_k^i)\} \cdot \mathbb{I}\{X_k^i = \mathrm{out}_i(U_k, \boldsymbol{T}_k^i)\}$.

ation, no a priori information is available thus the message $P_a(U_k)$ is initialized as a uniform probability mass function. The forward recursion α becomes

$$\alpha_{k+1}(\boldsymbol{T_{k+1}}) = \sum_{\sim\{\boldsymbol{T_{k+1}}\}} \alpha_k(\boldsymbol{T_k}) \cdot P(y_k^i|X_k^i) \cdot \mu_k(U_k) \times \mathbb{I}\{\boldsymbol{T_{k+1}} \leftarrow (U_k, \boldsymbol{T_k})\} \cdot \mathbb{I}\{X_k^i = \text{out}_C(U_k, \boldsymbol{T_k})\} \quad (2.41)$$

and the backward recursion β is

$$\beta_k(\boldsymbol{T_k}) = \sum_{\sim\{\boldsymbol{T_k}\}} \beta_{k+1}(\boldsymbol{T_{k+1}}) \cdot P(y_k^i|X_k^i) \cdot \mu_k(U_k) \times \mathbb{I}\{\boldsymbol{T_{k+1}} \leftarrow (U_k, \boldsymbol{T_k})\} \cdot \mathbb{I}\{X_k^i = \text{out}_C(U_k, \boldsymbol{T_k})\}. \quad (2.42)$$

From these, the decoder calculates so-called *extrinsic information* :

$$P_e(U_k) = \sum_{\sim\{U_k\}} \alpha_k(\boldsymbol{T_k}) \cdot \beta_{k+1}(\boldsymbol{T_{k+1}}) \cdot P(y_k^i|X_k^i) \times \mathbb{I}\{\boldsymbol{T_{k+1}} \leftarrow (U_k, \boldsymbol{T_k})\} \cdot \mathbb{I}\{X_k^i = \text{out}_C(U_k, \boldsymbol{T_k})\}. \quad (2.43)$$

The expression of these messages can be simplified by removing terms that the indicator functions cancel; this amounts to make implicit the correspondence between coded bits $\boldsymbol{X}$ and information bits $\boldsymbol{U}$. Defining

$$\gamma_k(U_k, \ldots, U_{k-M}) \triangleq P(y_k^i|X_k^i) \cdot P(y_k^0|U_k) \cdot P_a(U_k), \quad (2.44)$$

we can write

$$\alpha_{k+1}(U_k, \ldots, U_{k-M+1}) = \sum_{U_{k-M}} \alpha_k(U_{k-1}, \ldots, U_{k-M}) \cdot \gamma_k(U_k, \ldots, U_{k-M}) \quad (2.45)$$

$$\beta_k(U_{k-1}, \ldots, U_{k-M}) = \sum_{U_k} \beta_{k+1}(U_k, \ldots, U_{k-M+1}) \cdot \gamma_k(U_k, \ldots, U_{k-M}) \quad (2.46)$$

$$P_e(U_k) = \sum_{U_{k-1}, \ldots, U_{k-M}} \alpha_k(U_{k-1}, \ldots, U_{k-M}) \cdot \beta_{k+1}(U_k, \ldots, U_{k-M+1}) \cdot P(y_k^i|X_k^i). \quad (2.47)$$

Information exchanges between the constituent decoders As seen on the factor graph, the extrinsic information $P_e(U_k)$ computed by a constituent decoder is sent to the other through the interleaver, and used as a priori information $P_a(U_k)$.

The interpretation of the extrinsic information appears when calculating the marginal relative to U_k, i.e. the a posteriori probability of the bit U_k (up to a multiplicative constant, given the simplifications of the objective function). According to the rules of the sum-product algorithm, the approximate a posteriori probability is, after a sufficient number of iteration, the product of incoming messages at the node U_k; thus :

$$P(U_k|\boldsymbol{y}) \approx c\ P_a(U_k) \cdot P(y_k^0|U_k) \cdot P_e(U_k). \tag{2.48}$$

This probability mass function, used to find an estimate $\hat{U}_k$, combines three sources of information :

- a priori information on the bit U_k,
- the observation of the channel,
- and extrinsic information.

Since the first two quantities are provided to the constituent decoder, the extrinsic information can be interpreted as the additional information it produces.

Note that the calculation of extrinsic information $P_e(U_k)$ does not use information on the current bit U_k. Hence, the extrinsic information is not redundant with the a priori information and with the channel observation (this is rigorously true only with an infinite-length interleaver, or as long as no cycle has been completed on the graph). Moreover, the extrinsic information $P_e(U_k)$ sent to the other decoder does not depend on the a priori information $P_a(U_k)$ received from that decoder; this ensures not reusing known information as if it were fresh independent information (again as long as no cycle has been completed on the graph).

Summary of the turbo-decoding algorithm The block diagram of fig. 2.9 summarizes the turbo-decoding process.

After depuncturing and demultiplexing, the channel observations are transmitted to the constituent decoders. The first one computes its extrinsic information, using uniform a priori. The sequence of extrinsic information is then interleaved and sent as a priori information for the second decoder. The latter in turn computes its extrinsic information, which is recycled as a priori information by the first decoder, and so on. After several iterations, the approximation of $P(U_k|\boldsymbol{y})$ is calculated at the output

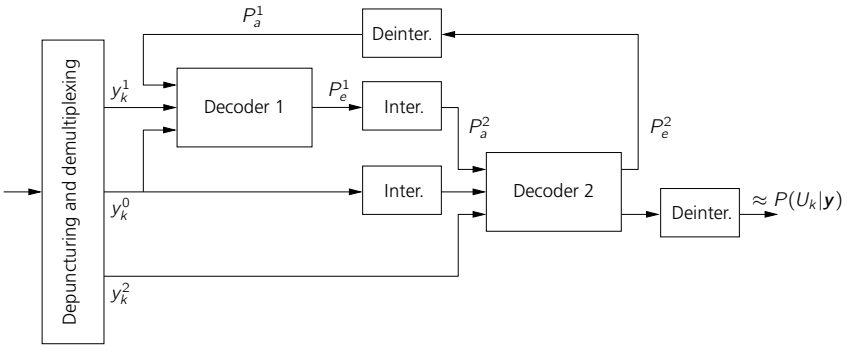


Figure 2.9: Block diagram of the turbo-decoder.

of any decoder by combining the channel observation with the decoder a priori and extrinsic informations.

The scheduling of the turbo-decoding algorithm can be illustrated on the factor graph, as in fig. 2.10. The correspondence with the previous block diagram is as follows. Operation (a) consists in providing the channel observations $P(y_k^i|X_k^i)$ to the decoders ($i = 1, 2$), and to send the message $P(y_k^0|U_k)$ to the nodes U_k. This step is carried out only once, since these messages do not change during the decoding process. Steps (b) to (e) are performed iteratively, in turns by each decoder (the figure considers the first decoder only). Figure (b) corresponds to sending the information available on U_k to the constituent decoder; this information consists of a priori information P_a^i and of information from the channel $P(y_k^0|U_k)$. Next, fig. (c) represents the forward recursion α_k^i, whereas fig. (d) represents the backward recursion β_k^i. Finally, step (e) corresponds to the computation of extrinsic information P_e^i of the decoder, or equivalently to the update of the a priori information of the other decoder. The process returns to step (b), but for the second decoder, and iterates. After several iterations, the product of incoming messages at the node U_k yields the approximation of $P(U_k|\boldsymbol{y})$.

This scheduling of operations yields an algorithm that uses the decoders of the constituent codes alternately. Other schedulings are possible, and do not necessarily make the constituent decoders apparent.

Comments

Effect of cycles The performance of the turbo-decoding algorithm is remarkable, even though the presence of cycles in the graph make it sub-optimal (in comparison with the MAP receiver). This favorable behavior is attributable to the presence of an

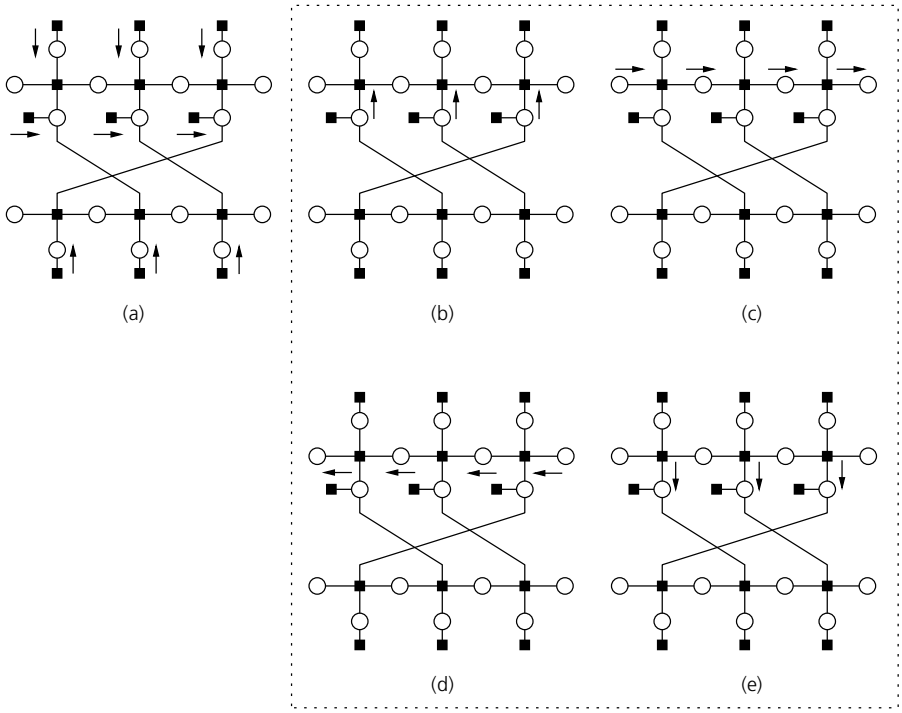


Figure 2.10: Illustration of the scheduling of the turbo-decoder, on the factor graph. Steps inside the dotted frame are carried out iteratively, aternately by each decoder.

interleaver, that lengthen cycles in the graph. Intuitively and without going into the details here, long cycles are less damaging than short ones because many iterations are required before messages complete a cycle; the first iterations are thus relatively less exposed to the effect of cycles. This emphasizes the importance of the design of good interleavers.

Implementation In practice, as for convolutional codes, implementations use a logarithmic formulation of the algorithm. Expressions for each constituent decoder are very similar to those for a convolutional code; they are not detailed here.

Lower complexity variants As for convolutional codes, a lower complexity variant of the turbo-decoding algorithm can be obtained by working in the max-sum semiring (as in section 2.2.3).

2.4 BIT-INTERLEAVED CODED MODULATION

Coded modulation consists in combining error-correcting codes with multilevel modulations to achieve various trade-offs between error performance an data transmission rate, for a fixed bandwidth.

The design of coded modulation with jointly optimized code and modulation has first been addressed by Ungerboeck [25, 26] and Imai [27]. In the particular case of fading channels however, Zehavi and Caire have shown that such joint optimization of coding and modulation is outperformed by a technique called bit-interelaved coded modulation (BICM), that better benefits from time diversity [28, 29]. A BICM encoder consists of a convolutional code, followed by a bit interleaver and a multilevel modulator. For BICM, the optimal modulation was found to be Gray mapping; it is thus not jointly optimized with the code. Later, Li and Ritcey have discovered that BICM can outperform joint design of coding and modulation over non-fading channels as well, when using an iterative receiver [30, 31]. The optimal modulation is no longer Gray mapping in that case, as analyzed in [32, 33].

The iterative receiver for BICM turns out to be another particular instance of the sum-product algorithm, as shown in this section. The first part of this section defines the BICM coding scheme, and the second part presents the derivation of the BICM iterative decoding algorithm in the factor graphs framework.

2.4.1 Definition and encoding

A BICM encoder is made up of a convolutional encoder, an interleaver and a multilevel modulator, as shown in fig. 2.11. The sequence of information bits $\boldsymbol{U}$ (length K) is encoded into a codeword $\boldsymbol{X}$ (length $n \cdot K$), interleaved and modulated into a sequence $\boldsymbol{S}$ of symbols (length $N = n \cdot K/m$, where 2^m is the size of the modulation alphabet).

The overall code rate is thus $R = \frac{m}{n}$.

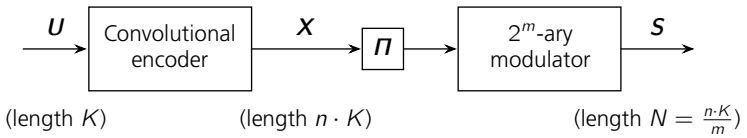


Figure 2.11: BICM encoder.

2.4.2 Decoding algorithm

Again, as seen in section 2.1, MAP decoding consists in solving the following maximization problem :

$$\begin{aligned} \hat{U}_k &= \arg \max_{U_k=-1,1} P(U_k|\boldsymbol{y}) \\ &= \arg \max_{U_k=-1,1} \sum_{\sim\{U_k\}} P(\boldsymbol{y}|\boldsymbol{S}) \cdot P(\boldsymbol{S}|\boldsymbol{X}) \cdot P(\boldsymbol{X}|\boldsymbol{U}). \end{aligned}$$

In the following we factorize the three factors above, then draw the factor graph and apply the sum-product algorithm.

Factorization of the objective function

Factorization of $P(\boldsymbol{y}|\boldsymbol{S})$ Assuming a memoryless channel model, this factor can be written :

$$P(\boldsymbol{y}|\boldsymbol{S}) = \prod_{i=1}^{N} P(y_i|S_i) \tag{2.49}$$

where y_i is the channel observation for the transmitted symbol S_i.

Factorization of $P(\boldsymbol{S}|\boldsymbol{X})$ This factor characterizes the correspondence between the codeword $\boldsymbol{X}$ and the symbols $\boldsymbol{S}$; it is equal to 1 only if $\boldsymbol{S} = \text{out}_M(\boldsymbol{X})$, and to 0

otherwise. For a modulator without memory we have

$$\begin{aligned} P(\boldsymbol{S}|\boldsymbol{X}) &= \mathbb{I}\{\boldsymbol{S} = \text{out}_M(\boldsymbol{X})\} \\ &= \prod_{i=1}^{N} \mathbb{I}\{S_i = \text{out}_M(X_i^1 \dots X_i^m)\} \end{aligned} \tag{2.50}$$

where $X_i^1 \dots X_i^m$ are the interleaved coded bits that are mapped on the symbol S_i.

Factorization of $P(\boldsymbol{X}|\boldsymbol{U})$ Finally, the factor $P(\boldsymbol{X}|\boldsymbol{U})$ characterizes the correspondence between the information sequence $\boldsymbol{U}$ and the codeword $\boldsymbol{X}$ produced by the convolutional encoder. We have, as for a convolutional code in equation (2.20),

$$P(\boldsymbol{X}|\boldsymbol{U}) = \mathbb{I}\{\boldsymbol{X} = \text{out}_C(\boldsymbol{U})\} \tag{2.51}$$

$$= \mathbb{I}\{\boldsymbol{T_1} = \mathbf{0}\} \prod_k \mathbb{I}\{\boldsymbol{T_{k+1}} \leftarrow (U_k, \boldsymbol{T_k})\} \tag{2.52}$$

$$\times\ \mathbb{I}\{X_k^1, \dots, X_k^n = \text{out}_C(U_k, \boldsymbol{T_k})\}.$$

Note that we use a different indexing for coded bits before and after interleaving; before interleaving we write them $X_k^1, \dots, X_k^n$ with $k = 1, \dots, K$, and after interleaving $X_i^1 \dots X_i^m$ with $i = 1, \dots, N$.

Factor graph

The factor graph corresponding to the previous factorization is depicted in fig. 2.12.

This graph contains cycles, made longer thanks to the interleaver. The sum-product algorithm will thus be iterative. Moreover, the graph logically contains the graph of a BCJR decoder; the BICM decoding algorithm will thus include the BCJR algorithm (not necessarily though : this depends on the scheduling).

Application of the sum product algorithm

Decoding of the convolutional code The decoding algorithm for the constituent code is the BCJR algorithm of section 2.2.2, behalves that it has to produce extrinsic information on the parity bits, i.e. the messages $P_e(X_k^1), \dots, P_e(X_k^n)$ on the graph.

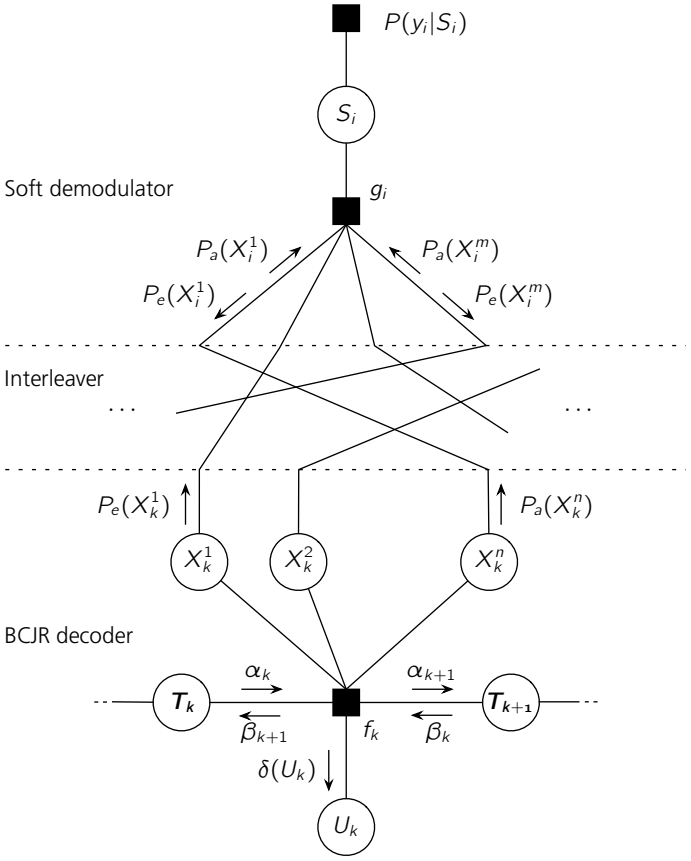


Figure 2.12: Section of the factor graph of the objective function for BICM decoding. The node f_k respresents the function $f_k = \mathbb{I}\{X_k^1, \ldots, X_k^n = \mathrm{out}_C(U_k, \boldsymbol{T_k})\}$, and the node g_i represents $g_i = \mathbb{I}\{S_i = \mathrm{out}_M(X_i^1 \ldots X_i^m)\}$.

They are equal to

$$\begin{aligned} P_e(X_k^q) &= \sum_{\sim\{X_k^q\}} \alpha_k(\boldsymbol{T_k}) \cdot \beta_{k+1}(\boldsymbol{T_{k+1}}) \cdot \left[\prod_{\substack{l=1 \\ l \neq q}}^{n} P_a(X_k^l) \right] \\ &\times \mathbb{I}\{\boldsymbol{T_{k+1}} \leftarrow (U_k, \boldsymbol{T_k})\} \\ &\times \mathbb{I}\{X_k^1 \ldots X_k^n = \mathrm{out}_C(U_k, \boldsymbol{T_k})\}, \end{aligned} \tag{2.53}$$

for $k = 1, \ldots, K$ and $q = 1, \ldots, n$. Keeping implicit the correspondence between information bits, coded bits and encoder states, yields equivalently

$$P_e(X_k^q) = \sum_{\sim\{X_k^q\}} \alpha_k(U_{k-1}, \ldots, U_{k-M}) \cdot \beta_{k+1}(U_k, \ldots, U_{k-M+1}) \cdot \prod_{\substack{l=1 \\ l \neq q}}^{n} P_a(X_k^l). \quad (2.54)$$

These extrinsic messages $P_e(X_k^1), \ldots, P_e(X_k^n)$ are then interleaved and used by the demodulator as a priori information $P_a(X_i^1), \ldots, P_a(X_i^m)$.

Demodulation The demodulator uses both the channel observations $\boldsymbol{y}$ and the a priori information from the decoder to produce messages of extrinsic information (messages $P_e(X_i^1), \ldots, P_e(X_i^m)$ on the graph). These messages are computed as follows :

$$P_e(X_i^p) = \sum_{\sim\{X_i^p\}} P(y_i|S_i) \cdot \mathbb{I}\{S_i = \text{out}_M(X_i^1 \ldots X_i^m)\} \cdot \prod_{\substack{j=1 \\ j \neq p}}^{m} P_a(X_i^j). \quad (2.55)$$

for $i = 1, \ldots, N$ and $p = 1, \ldots, m$. Equivalently, the more compact formulation with implicit correspondence between coded bits and symbols is

$$P_e(X_i^p) = \sum_{\sim\{X_i^p\}} P(y_i|S_i) \cdot \prod_{\substack{j=1 \\ j \neq p}}^{m} P_a(X_i^j). \quad (2.56)$$

Such a demodulator is referred to as a *soft demodulator* because it produces probabilities on the bits $X_i^1, \ldots, X_i^m$ rather than taking hard decisions on them.

Summary of the iterative BICM receiver The block diagram in fig. 2.13 schematizes the operations of the iterative BICM receiver. As shown above, it is made up of a BCJR decoder and of a demodulator, connected through an interleaver and a de-interleaver. The extrinsic messages produced by each block are used as a priori messages by the other. Finally the BCJR decoder produces approximations of the marginal probabilities $P(U_k|\boldsymbol{y})$, to take decisions $\hat{\boldsymbol{U}}$.

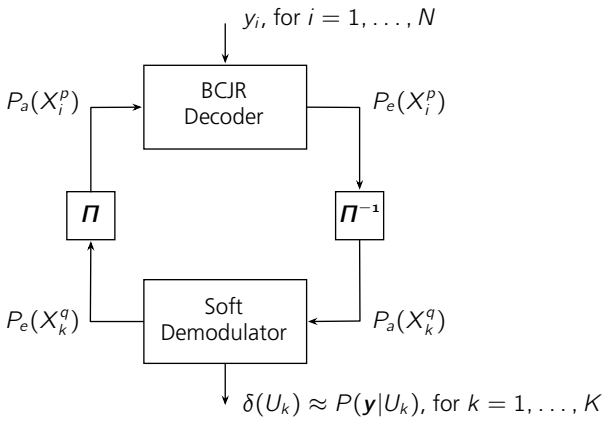


Figure 2.13: Block diagram of the iterative BICM receiver.

2.5 ANALYSIS OF ITERATIVE RECEIVERS

Analyzing the behavior of a receiver can provide information over the performance of the whole transmission system.

This section presents tools for analyzing the turbo-decoder : first EXIT charts [34, 35, 36], and then an analytical method with a similar purpose [1, 2]. Besides giving some insight on the behavior of the receiver itself, both these methods enable to characterize the performance of turbo-codes. More specifically, they enable to find the so-called *waterfall*, a threshold of channel reliability from which the bit error rate of turbo-codes decreases steeply.

Results are presented assuming transmission over an additive white Gaussian noise (AWGN) channel.

2.5.1 System model

The considered turbo-code is made up of two RSC codes separated by an interleaver in a typical parallel configuration (as in section 2.3). As these analysis tools characterize the two constituent codes separately, this section defines the notations for one of them only.

Encoding For each information bit U_k ($k = 1, \ldots, K$), each constituent code produces m parity bits X_k^l ($l = 1, \ldots, m$), as in section 2.2.

Modulation The information and parity bits are then modulated into BPSK symbols. Bits U_k, X_k^l are mapped to symbols S_k^0, $S_k^l \in \{1, -1\}$, respectively.

Channel The symbols are transmitted over an AWGN channel. The baseband-equivalent discrete-time signal is, for $k = 1, \ldots, K$:

$$\begin{array}{rcll} Y_k^0 & = & S_k^0 + Z_k^0, & \\ Y_k^l & = & S_k^l + Z_k^l, & l = 1, \ldots, n \end{array} \tag{2.57}$$

where Z_k^0, Z_k^l are noise samples from a zero-mean white Gaussian process with two-sided power spectral densitiy $N_0/2$. The total energy spent for transmitting one information bit is written E_b.

Demodulation From channel observations $Y_k^0 = y_k^0$ and $Y_k^l = y_k^l$, the demodulator calculates the likelihoods of the received symbols as follows :

$$P(y_k^0|U_k) = \frac{1}{\sqrt{\pi N_0}} \exp\left(-\frac{(y_k^0 - S_k^0)^2}{N_0}\right) \tag{2.58}$$

$$P(y_k^l|X_k^l) = \frac{1}{\sqrt{\pi N_0}} \exp\left(-\frac{(y_k^l - S_k^l)^2}{N_0}\right) \tag{2.59}$$

where the correspondance between bits U_k, X_k^l and symbols S_k^0, S_k^l is assumed implicitly.

2.5.2 EXIT Charts

Extrinsic information transfer charts (EXIT charts) analyze the behavior of an iterative receiver by observing the exchanges of extrinsic information between the constituent decoders. They describe the whole receiver by using separate input-output characterizations of each constituent decoder.

The first part of this section focusses on the input-output characterization of one constituent decoder, and the second part on the analysis of the whole receiver. Notations assume a parallel turbo-code, but the method can be used with serial turbo-codes as well.

Extrinsic information transfer functions

This section analyzes the input-output relationship of a constituent decoder, in terms of mutual information. More precisely, we are interested in the relationship between the mutual informations (indices k are omitted)

$$I_a \triangleq I(U; L_a) \qquad \text{and} \qquad I_e \triangleq I(U; L_e) \tag{2.60}$$

taken at the input and at the output of a constituent decoder respectively, where

$$L_a = \ln \frac{P_a(U=+1)}{P_a(U=-1)} \tag{2.61}$$

and similarly for L_e.

Analytical treatment of the input-output extrinsic transfer function of a convolutional decoder is difficult, so we resort to numerical simulations to compute it. This empirical approach is made possible by the two following observations, made in [35] :

1. First, $P(L_e|U)$, the distribution of the extrinsic log-ratios L_e given U at the output of a constituent decoder approaches a Gaussian-like distribution with increasing number of iterations.
2. Second, the interleaver ensures that the log-ratios L_a remain locally mutually uncorrelated, and uncorrelated with the channel observations, over many iterations.

For these reasons, one can generate input sequences of extrinsic log-ratios as if they were made of independent Gaussian-distributed log-ratios (it enables separate analysis of the constituent decoders because extrinsic log-ratios can be generated easily without taking the other constituent decoder into account). One determines the input-output correspondence between I_a and I_e by computing the empirical mutual information observed at the output, for each value of the mutual information fed at the input.

With the assumptions above, the a priori information L_a at the input of a constituent decoder can be modeled as

$$L_a = \mu_a U + Z_a \tag{2.62}$$

where Z_a is a centered white Gaussian noise with variance σ_a^2. Since L_a is assumed to be Gaussian-distributed (given U), the parameter μ_a must fulfill

$$\mu_a = \frac{\sigma_a^2}{2}. \tag{2.63}$$

The probability density function of L_a given U is then

$$P(L_a = \xi | U = u) = \frac{1}{\sqrt{2\pi}\sigma_a} e^{-\frac{(\xi - \frac{\sigma_a^2}{2} \cdot u)^2}{2\sigma_a^2}}. \tag{2.64}$$

The information contents of the sequence of log-ratios L_a is measured by the mutual information I_a between the transmitted bits U and the log-ratios :

$$\begin{aligned} I_a &= I(U; L_a) \\ &= \frac{1}{2} \sum_{u=-1,1} \int_{-\infty}^{+\infty} P(L_a = \xi | U = u) \\ &\times \log_2 \frac{2 \cdot P(L_a = \xi | U = u)}{P(L_a = \xi | U = -1) + P(L_a = \xi | U = +1)} \, \mathrm{d}\xi \end{aligned} \tag{2.65}$$

with the property that $0 \leq I_a \leq 1$. Substituting (2.64) yields :

$$I_a(\sigma_a) = \int_{-\infty}^{+\infty} \frac{e^{-\frac{(\xi - \frac{\sigma_a^2}{2})^2}{2\sigma_a^2}}}{\sqrt{2\pi}\sigma_a} \cdot (1 - \log_2(1 + e^{-\xi})) \, \mathrm{d}\xi, \tag{2.66}$$

along with the following definitions :

$$\lim_{\sigma \to 0} I_a(\sigma_a = \sigma) = 0, \quad \lim_{\sigma \to \infty} I_a(\sigma_a = \sigma) = 1 \tag{2.67}$$

$$\sigma_a > 0. \tag{2.68}$$

Similarly, the information contents of the sequence of log-ratios L_e at the output of the decoder are quantified by the mutual information I_e. Because the other constituent decoder receives the sequence of log-ratios L_e after interleaving, it regards this sequence as memoryless; consequently the calculation of the mutual information I_e uses the same assumption :

$$\begin{aligned} I_e &= I(U; L_e) \\ &= \frac{1}{2} \sum_{u=-1,1} \int_{-\infty}^{+\infty} P(L_e = \xi | U = u) \\ &\times \log_2 \frac{2 \cdot P(L_e = \xi | U = u)}{P(L_e = \xi | U = -1) + P(L_e = \xi | U = +1)} \, \mathrm{d}\xi \end{aligned} \tag{2.69}$$

where $P(L_e = \xi | U = u)$ is obtained empirically by observing the outputs L_e of the decoder, and considering them as independent and identically distributed. We again

have the property that $0 \leq I_e \leq 1$.

The mutual information I_e can be regarded as a function of the mutual information I_a and of the signal-to-noise ratio E_b/N_0; this defines the extrinsic information transfer function (EXIT function) :

$$I_e = T(I_a, E_b/N_0). \tag{2.70}$$

In practice, the transfer function $T(I_a, E_b/N_0)$ is obtained by simulation as follows : for a given E_b/N_0 and a value of I_a fixed by σ_a, transmitted bits U are decoded by using both channel observations and a priori information L_a generated according to (2.64). The empirical distribution $P(L_e|U)$ of the outputs of the decoder is measured in an histogram, and finally the mutual information I_e is calculated with (2.69).

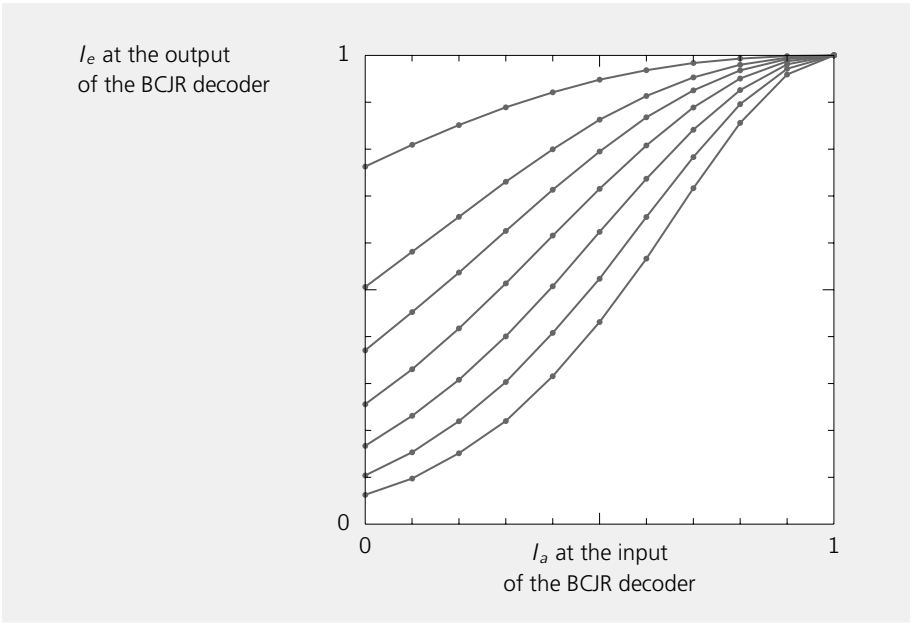


Figure 2.14: EXIT function for a RSC code with polynomials $(23_8, 37_8)$. Curves for $E_b/N_0 = -0.5, 0, 0.5, 1, 1.5, 2, 3$ dB (in increasing order).

Example 2.3 : Extrinsic transfer functions

Figure 2.14 represents the EXIT function of the RSC code with polynomials $(23_8, 37_8)$ for several values of the E_b/N_0 ratio. Predictably enough, the extrinsic mutual information I_e increases with the a priori information I_a. The extrinsic information I_e is non-zero even when $I_a = 0$ since the decoder also uses the channel observations,

besides a priori information. As expected, the higher the E_b/N_0 ratio the higher the extrinsic information. When $I_a = 1$, this code also has $I_e = 1$. ■

Trajectories of iterative decoding

One can visualize the trajectories of iterative decoding in a single diagram displaying the transfer functions of the two constituent decoders, by swapping the axes for the function of the second decoder. Such a diagram is commonly called an *EXIT chart*.

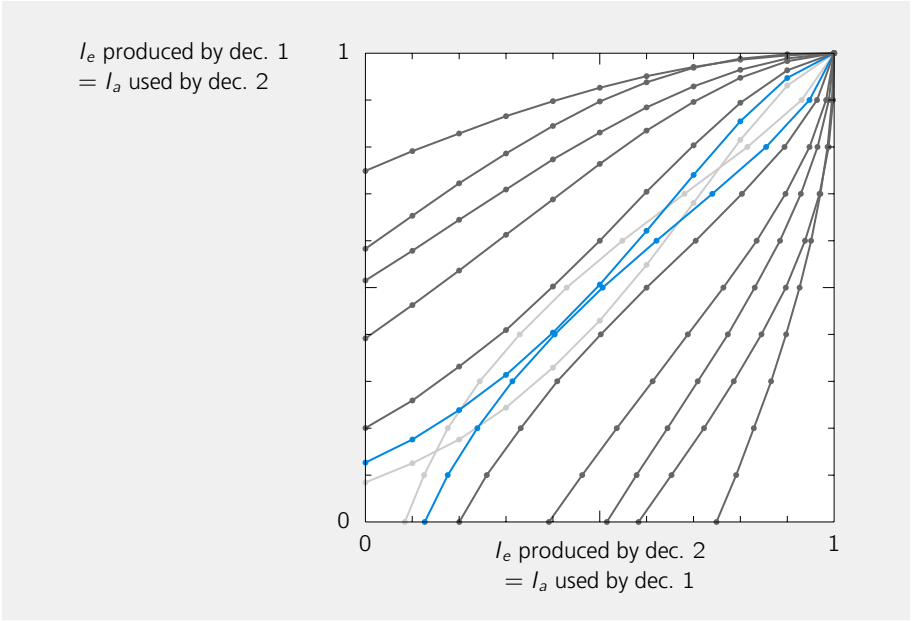


Figure 2.15: EXIT chart for the turbo-code made of two RSC constituent codes $(13_8, 15_8)$. Curves for $E_b/N_0 = -0.5$ dB (light gray), -0.05 dB (blue), 0.5, 1, 1.5, 2, 3 dB (dark gray).

Typical EXIT charts are plotted in fig. 2.15–2.17, where the EXIT functions of both decoders are displayed for several values of E_b/N_0. The transfer functions of the two constituent decoders are referred to as $T_1(I_{a,1}, E_b/N_0)$ and $T_2(I_{a,2}, E_b/N_0)$ in the following. Before the first iteration the a priori information is zero, thus the trajectory starts from the lower left corner where $I_{a,1}^{(0)} = 0$. Next, for every iteration ℓ, the extrinsic mutual information at the output of the first decoder is $I_{e,1}^{(\ell)} = T_1(I_{a,1}^{(\ell)}, E_b/N_0)$, and becomes the a priori mutual information for the second decoder : $I_{a,2}^{(\ell)} = I_{e,1}^{(\ell)}$. Using this a priori information the decoder computes an extrinsic mutual information $I_{e,2}^{(\ell)} = T_2(I_{a,2}^{(\ell)}, E_b/N_0)$, that becomes the a priori of the first decoder, for the next iteration :

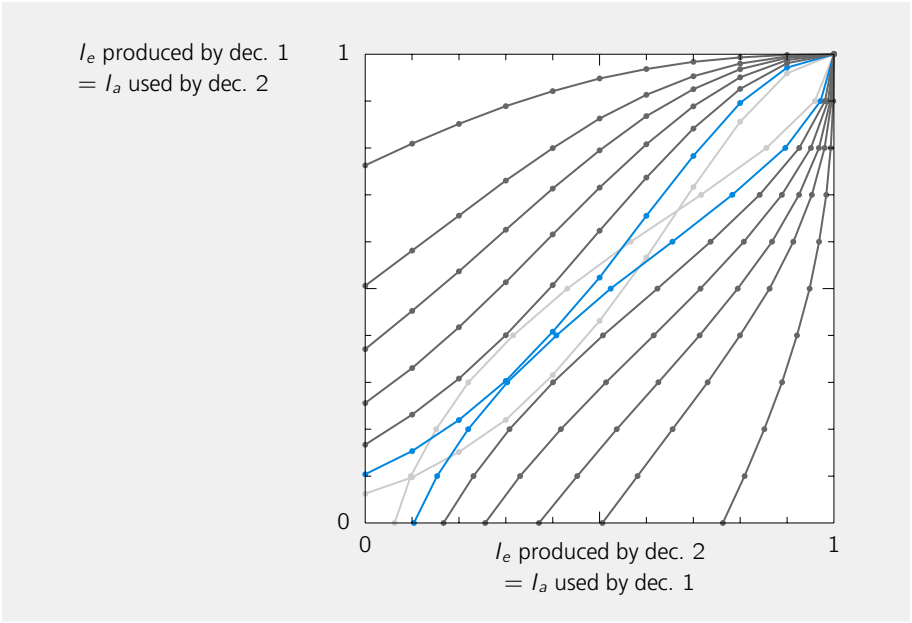


Figure 2.16: EXIT chart for the turbo-code made of two RSC constituent codes $(23_8, 37_8)$. Curves for $E_b/N_0 = -0.5$ dB (light gray), 0 dB (blue), 0.5, 1, 1.5, 2, 3 dB (dark gray).

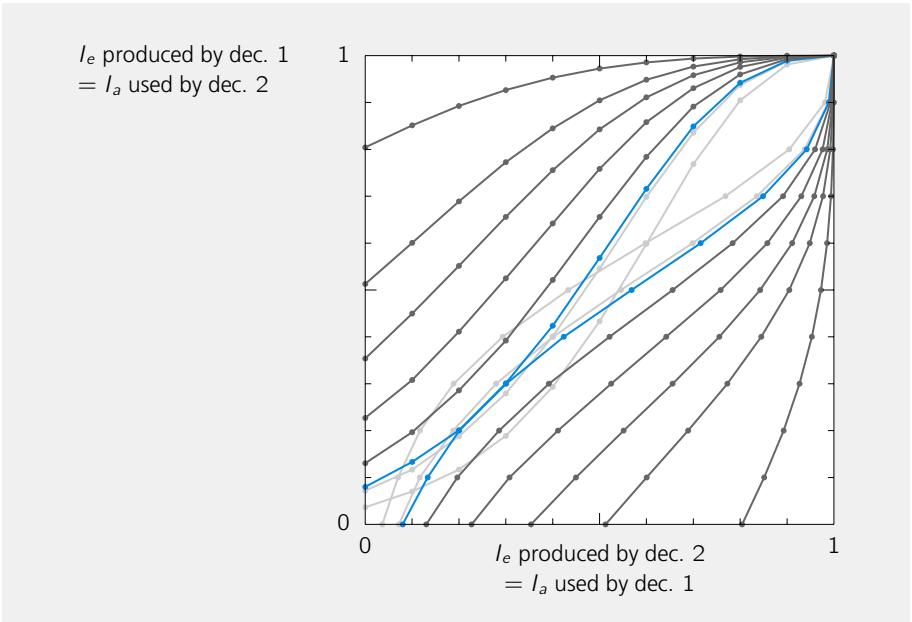


Figure 2.17: EXIT chart for the turbo-code made of two RSC constituent codes $(67_8, 45_8)$. Curves for $E_b/N_0 = -0.5, 0$ dB (light gray), 0.1 dB (blue), 0.5, 1, 1.5, 2, 3 dB (dark gray).

$I_{a,1}^{(\ell+1)} = I_{e,2}^{(\ell)}$, and so on.

Iterations continue as long as $I_{e,2}^{(\ell+1)} > I_{e,2}^{(\ell)}$, otherwise if $I_{e,2}^{(\ell+1)} = I_{e,2}^{(\ell)}$ the decoder gets stuck in a fixed point. This happens if the transfer functions T_1 et T_2 have an intersection : the decoder cannot reach a value of the extrinsic mutual information higher than the intersection, for this value of E_b/N_0. Successful decoding (asymptotically error-free) is achieved if the trajectory ends in the upper right point of the chart, where $I_{e,1} = I_{e,2} = 1$.

In fig. 2.15–2.17, the blue EXIT functions correspond to E_b/N_0 equal to the error-free decoding threshold, the lowest E_b/N_0 at which the EXIT functions have no intersection. The light gray curves correspond to situations that do not enable error-free decoding (the EXIT functions cut each other), and the dark gray to situations where error-free decoding is possible.

In practice, real decoding trajectories follow the predicted trajectory if the interleaver is large enough, because the assumptions required by the EXIT charts method stay satisfied over many iterations. For small interleavers, the real trajectory becomes lower than the predicted one after some iterations.

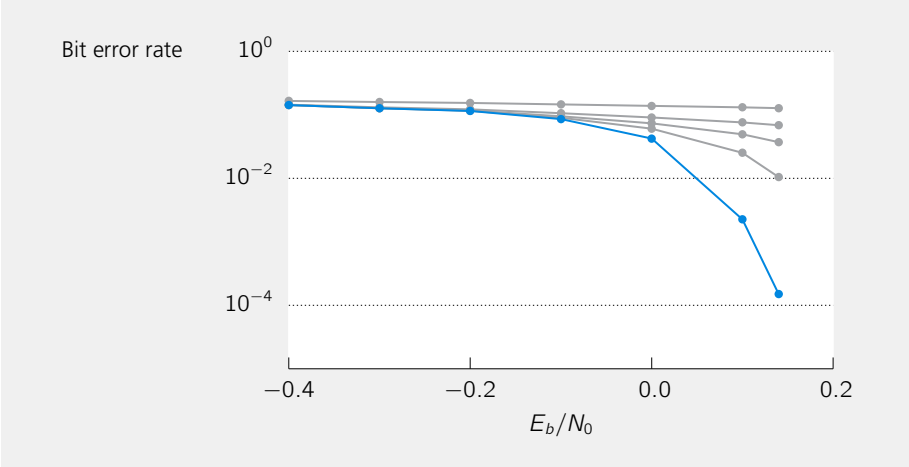


Figure 2.18: Bit error rate for the turbo-code made of two RSC constituent codes $(23_8, 37_8)$. Blocks of 65536 information bits, curves for 1, 3, 5, 7 and 10 iterations (from top to bottom).

For example, the error-free decoding threshold for the code of fig. 2.15 is equal to $E_b/N_0 = -0.05$ dB. The bit error-rate achieved with this turbo-code on blocks of 65536 information bits is plotted in fig. 2.18, for decoding in 1, 3, 5, 7 and 10 itera-

tions. The waterfall is clearly visible; it starts indeed close to $E_b/N_0 = -0.05$ dB. The longer the sequence, the steeper the waterfall.

2.5.3 Analytical prediction of the convergence threshold for turbo-codes

In contrast with the EXIT charts method, based on functions obtained by simulation, this section addresses the analytical calculation of performance characteristics of turbo-codes. The aim is to provide a method that has a low complexity and that enables the analysis of a wide variety of turbo-coded schemes, with in mind the possibility of using it as an optimization criterion in an adaptive system.

The proposed method enables the determination of the error-free decoding threshold and of the number of iterations to achieve error-free decoding. Its complexity is low and it offers an accuracy close the one of EXIT charts.

From the description of the turbo-decoding algorithm in section 2.3, we now derive the new characterization of turbo-codes performance.

The analysis carried out here is based on the evolution of the average extrinsic information throughout iterations. For SNRs above the error-free decoding threshold, the extrinsic information exchanged between the constituent decoders becomes ambivalent, i.e. high for the correct value of the information bit U_k and close to 0 for the other. The analysis is based on the expectation that average values of the extrinsic probabilities will reflect this behavior.

The first section below details the calculation of the *average values* of $P_e(U_k)$ for a convolutional code, assuming BCJR decoding. The second one addresses the case of turbo-codes.

Average extrinsic information for RSC codes

This section describes the calculation of average extrinsic information produced by a constituent decoder. The extrinsic information, given in (2.43), is here slightly modified to account for the more general case of m parity bits per information bit, instead of 1 in (2.43); this yields :

$$P_e(U_k) = \sum_{\sim\{U_k\}} \alpha_k(\boldsymbol{T_k}) \cdot \beta_{k+1}(\boldsymbol{T_{k+1}}) \cdot \prod_{l=1}^{m} P(y_k^l | X_k^l), \tag{2.71}$$

where the correspondence between states $\boldsymbol{T_k}$, $\boldsymbol{T_{k+1}}$ and bits U_k, X_k^l is kept implicit.

We assume that an infinite-length all-one codeword has been transmitted (because of the linearity of RSC codes, this is equivalent from a performance point of view to the transmission of any other code word). A priori information and channel observations are assumed to be mutually independent, as for EXIT charts. We are interested in the expected value of the extrinsic information at the output of the convolutional decoder for any information bit U_k. The expectation is over the a priori information P_a and over the channel observations, i.e. the realizations of the noise, for a given SNR.

The calculation of the average extrinsic information can be broken down in the calculation of the averages of its factors, because these are independent. As it can be seen from equation (2.71), each term in the expression of the extrinsic information is compound of three statistically independent factors. The first factor, α_k, depends on the set of channel observations preceding the k^{th} bit; the second factor, β_{k+1}, depends on the set of channel observations following the k^{th} bit; and finally the remaining factor depends only on the channel observations for the parity bits X_k^l, with $l = 1, \ldots, m$. Therefore when averaging over noise realizations, the average of the product of the three factors is the product of the averages of each factor.

Average of channel observations We first calculate the conditional average likelihood of the channel observations corresponding to the parity bits, knowing that $X_k^l = 1$ has been sent. This average likelihood is :

$$\begin{aligned}\overline{P}(X_k^l) &\triangleq \mathrm{E}\left\{P(y_k^l|X_k^l) \mid X_k^l = 1\right\} &(2.72)\\ &= \int_{-\infty}^{\infty} P(y_k^l|X_k^l)\; P(y_k^l|X_k^l = 1)\; dy_k^l \\ &= \begin{cases} \frac{1}{\sqrt{2\pi N_0}} & \text{if } U_k = 1 \\ \frac{1}{\sqrt{2\pi N_0}}\, e^{-2/N_0} & \text{if } U_k = -1. \end{cases} &(2.73)\end{aligned}$$

Average of α_k and β_k Next we consider the calculation of the averages of the functions α_k and β_{k+1}, that we write $\overline{\alpha}_k$ and $\overline{\beta}_{k+1}$. This can be stated as a linear eigenvalue problem. To formulate (2.41) and (2.42) in matrix notation, we form the vectors $\boldsymbol{\alpha}_k$ and $\boldsymbol{\beta}_{k+1}$ by stacking the functions $\alpha_k(\boldsymbol{t}_k)$ and $\beta_{k+1}(\boldsymbol{t}_{k+1})$ in a vector by

ascending values of their argument

$$\boldsymbol{\alpha}_k[\boldsymbol{T_k}] = \alpha_k(\boldsymbol{T_k}) \quad (2.74)$$

$$\boldsymbol{\beta}_{k+1}[\boldsymbol{T_{k+1}}] = \beta_{k+1}(\boldsymbol{T_{k+1}}). \quad (2.75)$$

If the considered constituent code has a memory of length M, the length of these vectors is 2^M. Then the recursions (2.41) and (2.42) can be rewritten

$$\boldsymbol{\alpha}_{k+1} = \boldsymbol{\Gamma}_k \cdot \boldsymbol{\alpha}_k \quad (2.76)$$

$$\boldsymbol{\beta}_k = \boldsymbol{\Gamma}_k^T \cdot \boldsymbol{\beta}_{k+1}, \quad (2.77)$$

where the definition of the $2^M \times 2^M$ transition matrix $\boldsymbol{\Gamma}_k$ follows :

$$\boldsymbol{\Gamma}_k[\boldsymbol{T_k}, \boldsymbol{T_{k+1}}] = \gamma_k\big(U_k, \ldots, U_{k-M}\big) \quad (2.78)$$

for $U_k, \ldots, U_{k-M}$ corresponding to $\boldsymbol{T_k}, \boldsymbol{T_{k+1}}$, and with $\gamma_k\big(U_k, \ldots, U_{k-M}\big)$ adapted from 2.44 to account for the m parity bits per information bit (instead of one in 2.44) :

$$\gamma_k(U_k, \ldots, U_{k-M}) \triangleq P(y_k^0|U_k) \cdot P_a(U_k) \prod_{l=1}^{m} P(y_k^l|X_k^l). \quad (2.79)$$

In the following we consider only the case of $\boldsymbol{\alpha}_{k+1}$, the case of $\boldsymbol{\beta}_k$ is identical. The resolution of the recursion (2.76) gives

$$\boldsymbol{\alpha}_{k+1} = \boldsymbol{\Gamma}_k \cdot \boldsymbol{\Gamma}_{k-1} \ldots \boldsymbol{\Gamma}_0 \cdot \boldsymbol{\alpha}_0 \quad (2.80)$$

where $\boldsymbol{\alpha}_0$ is the appropriate initialization of the recurrence. All the matrices in the product are mutually independent since they are functions of independent channel observations and because of the independence assumption of the a priori information. The average of this product of independent transition matrices is thus the product of the averages; moreover they all have the same average $\overline{\boldsymbol{\Gamma}}$ since they have the same probability distribution. By taking the average of (2.80) it follows that

$$\overline{\boldsymbol{\alpha}}_{k+1} = \overline{\boldsymbol{\Gamma}}^{k+1} \cdot \overline{\boldsymbol{\alpha}}_0 \quad (2.81)$$

where $\overline{\boldsymbol{\Gamma}}$ is the average transition matrix :

$$\begin{aligned}\overline{\boldsymbol{\Gamma}}[\boldsymbol{t}_{k+1}, \boldsymbol{t}_k] &= \overline{\gamma}_k(U_k, \ldots, U_{k-M}) \qquad (2.82)\\ &= \overline{P}(U_k)\, \overline{P}_a(U_k) \prod_{l=1}^{m} \overline{P}(X_k^l) \qquad (2.83)\end{aligned}$$

with $\overline{P}(U_k)$ defined and calculated as $\overline{P}(X_k^l)$ in (2.72).

Going on with our development from equation (2.81), we note that for sufficiently large values of k, the vector $\overline{\boldsymbol{\alpha}}_{k+1}$ is proportional to the eigenvector $\boldsymbol{v}_\alpha$ corresponding to the dominant eigenvalue λ of $\overline{\boldsymbol{\Gamma}}$:

$$\overline{\boldsymbol{\alpha}}_{k+1} \propto \boldsymbol{v}_\alpha. \qquad (2.84)$$

Vectors $\overline{\boldsymbol{\alpha}}_k$ for successive values of k differ by a factor λ. A similar result holds for the recurrence $\boldsymbol{\beta}_k$, and we note $\boldsymbol{v}_\beta$ the dominant eigenvector of $\overline{\boldsymbol{\Gamma}}^T$ (the dominant eigenvalues of $\overline{\boldsymbol{\Gamma}}$ and $\overline{\boldsymbol{\Gamma}}^T$ are equal).

Average extrinsic information Finally, combining (2.79), (2.71) and (2.84) enables to calculate the average extrinsic information, which is the same for every k :

$$\overline{P}_e(U_k) \propto \sum_{\sim U_k} \boldsymbol{v}_\alpha\big[\boldsymbol{T_k}\big] \cdot \boldsymbol{v}_\beta\big[\boldsymbol{T_{k+1}}\big] \cdot \prod_{l=1}^{m} \overline{P}(X_k^l), \qquad (2.85)$$

again for values of the bits X_k^l compatible with the state transition from $\boldsymbol{T_k}$ to $\boldsymbol{T_{k+1}}$. The proportionality factor between the two sides of the equality in (2.85) depends on the initialization of the recursions and on the eigenvalue λ to the power K. This proportionality factor changes with K but the ratio between $\overline{P}_e(U_k = 1)$ and $\overline{P}_e(U_k = -1)$ stays the same. The generalization to an infinite-length code word ($K \to \infty$) is straightforward.

Note that $P_e(U_k)$ is not defined as a probability since it is not normalized to 1; it is instead a likelihood measure as are the outputs of the demodulator (2.58) and (2.59). As a consequence, the averages are not normalized either.

Average extrinsic information for turbo-codes

The analysis above is easily generalized to the case of turbo-codes. The assumption of independence between the a priori information and the channel observations is supported by the infinite-length code word assumption. Indeed, it places us in the context of large interleavers for which the recycled a priori information remains roughly uncorrelated, locally, from the corresponding channel observations.

Given the result (2.85) one can calculate the evolution of the average extrinsic probability $\overline{P}_e$ along the whole iterative decoding process. Indeed equation (2.85) establishes, for a fixed SNR on the channel, the relationship between the average a priori information $\overline{P}_a$ (through $\boldsymbol{v}_\alpha$ and $\boldsymbol{v}_\beta$) at the input of a constituent decoder and the average extrinsic information $\overline{P}_e$ at its output. At the first iteration and for the first constituent decoder the average extrinsic information $\overline{P}_e$ is calculated assuming equiprobable a priori information at its input, so that $\overline{P}_a(U_k)$ is equal to $1/2$ for both values of the bit U_k. At the following iterations the average a priori information is equal to the average extrinsic information of the preceding step. Error-free decoding will be assumed in cases where $\overline{P}_e(U_k = 1)$ converges to zero.

This sequence of average extrinsic information can be visualized in a chart, yielding something similar to the trajectories of mutual information in an EXIT chart. Figure 2.19 is an example of such $\overline{P}_e$-trajectories. After calculating the sequence of average extrinsic information $\overline{P}_e$ as above, the values of $\log(\overline{P}_e(U_k = +1) \,/\, \overline{P}_e(U_k = -1))$ have been plotted in a diagram with the input of the first decoder / output of the second as abscissa and the output of the first decoder / input of the second as ordinate. Trajectories that reach error-free decoding go to infinity since $\overline{P}_e(U_k = -1)$ gets equal to zero; on the contrary trajectories corresponding to SNRs below the error-free decoding threshold converge to a finite value.

Accuracy of the predictions

The calculated code properties compare well to the more accurate EXIT charts predictions, as we will see below. The reason for comparing to the EXIT charts method is that these charts enable the evaluation of the same decoding properties : the error-free decoding threshold for infinite-length code words and, for signal-to-noise ratios above that decoding threshold, the number of decoding iterations necessary to achieve error-free decoding.

The differences between calculated code properties and the predictions of the EXIT

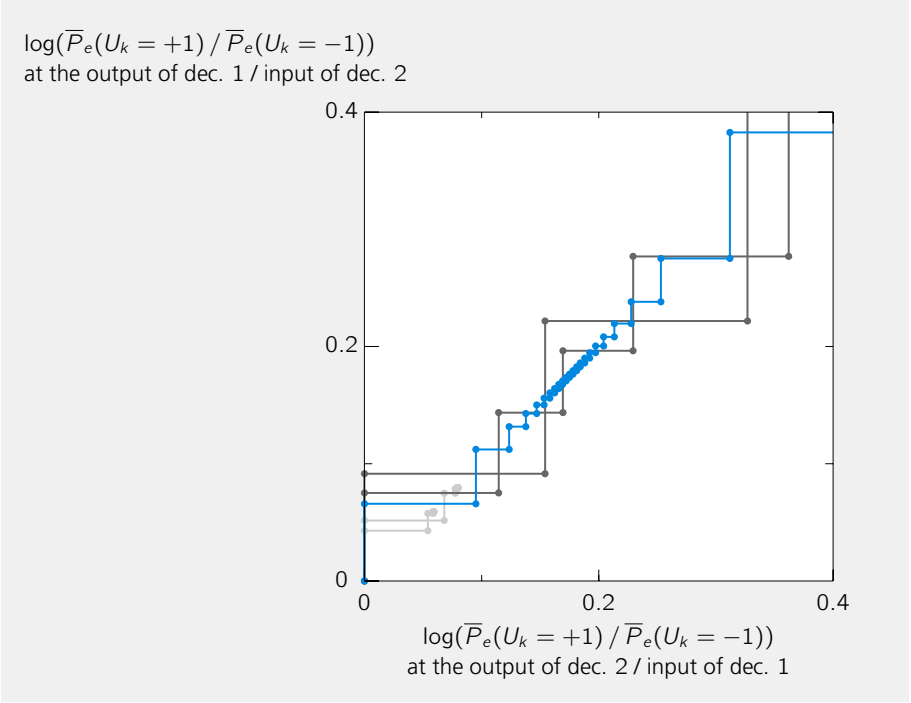


Figure 2.19: Visualization of the evolution of $\overline{P}_e$ for the turbo-code based on two RSC code with polynomials $(23_8, 37_8)$. The two trajectories in the lower left corner (light gray) do not enable error-free decoding; they correspond to $E_b/N_0 = -0.2$ and -0.1 dB. The other three trajectories continue outside the range of the figure and finally go to infinity; they enable error-free decoding. The blue trajectory corresponds to $E_b/N_0 = 0.032$ dB, the lowest E_b/N_0 found that enables error-free decoding. The two dark gray trajectories correspond to $E_b/N_0 = 0.1$ and 0.2 dB respectively.

charts method are due to the finite precision of the calculations. EXIT charts rely on an average mutual information measure, which is more robust than the one we propose; as a consequence the threshold predictions are more accurate. The slightly lower accuracy of the method presented above is the price for the much lower complexity.

The comparisons are carried out for a turbo-code sent on an AWGN channel, with BPSK modulation, and the same signal-to-noise ratio on all coded bits. A comparison of the threshold for error-free decoding evaluated with the EXIT charts method to those calculated as above is presented in table 2.1, for three turbo-codes. As shown in the table, calculated thresholds are in accordance with the evaluations of the EXIT charts method, even though they are slightly less accurate. Table 2.2 contains a comparison of the number of iterations of turbo-decoding, evaluated with EXIT charts or calculated.

Both methods have very similar predictions.

Code	EXIT charts	Calculated
$(13_8, 15_8)$	-0.05 dB	-0.1 dB
$(23_8, 37_8)$	0 dB	0.032 dB
$(67_8, 45_8)$	0.1 dB	0.3 dB

Table 2.1: Error-free decoding thresholds, expressed in terms of E_b/N_0. The first column is the polynomial of the constituent RSC codes making up the turbo-code, in octal notation (they have memories of length 3, 4 and 5 respectively, and all three turbo-codes have a rate 1/3).

	$(13_8, 15_8)$		$(23_8, 37_8)$		$(67_8, 45_8)$	
E_b/N_0	EXIT	Calc.	EXIT	Calc.	EXIT	Calc.
0.5 dB	6	6	5	5	5	5
1 dB	5	5	4	4	3	3
2 dB	3	4	2	3	2	2
3 dB	2	3	2	2	2	2

Table 2.2: For the same turbo-codes as in table 2.1, the number of iterations of turbo-decoding necessary to achieve error-free decoding. In each column, the left part presents the evaluations made with EXIT charts and the right part the calculated number of iterations.

Appendix 2.A LOGARITHMIC FORMULATION OF THE BCJR ALGORITHM

The BCJR algorithm suffers from numerical accuracy problems when implemented as formulated in section 2.2. These accuracy problems come from the difference in order of magnitude between the probabilities involved; they arise especially when the BCJR algorithm is used in a turbo-decoder.

To prevent these numerical accuracy problems, implementations preferably use the logarithmic formulation of the algorithm detailed below.

The objective of the BCJR algorithm is to compute a function $\delta_k(U_k)$, proportional to the a posteriori probability mass function $P(U_k|\boldsymbol{y})$. The corresponding log-ratio is

$$L_{U,k} = \ln\left(\frac{\delta_k(U_k = +1)}{\delta_k(U_k = -1)}\right). \tag{2.86}$$

Developing this expression by using (2.30) yields

$$L_{U,k} = \ln\left(\frac{\sum_{\sim\{U_k=+1\}} \alpha_k \cdot \beta_{k+1} \cdot \gamma_k}{\sum_{\sim\{U_k=-1\}} \alpha_k \cdot \beta_{k+1} \cdot \gamma_k}\right). \tag{2.87}$$

Notations are kept to the minimum here, section 2.2 gives the necessary details if needed.

The logarithmic formulation of the BCJR algorithm deals with functions α_k, β_{k+1} and γ_k by using their logarithm written $\bar{\alpha}_k = \ln(\alpha_k)$, $\bar{\beta}_{k+1} = \ln(\beta_{k+1})$ and $\bar{\gamma}_k = \ln(\gamma_k)$ respectively. Expression (2.87) becomes

$$\begin{aligned} L_{U,k} &= \ln\left(\frac{\sum_{\sim\{U_k=+1\}} \exp(\bar{\alpha}_k) \cdot \exp(\bar{\beta}_{k+1}) \cdot \exp(\bar{\gamma}_k)}{\sum_{\sim\{U_k=-1\}} \exp(\bar{\alpha}_k) \cdot \exp(\bar{\beta}_{k+1}) \cdot \exp(\bar{\gamma}_k)}\right) \\ &= \ln\left(\sum_{\sim\{U_k=+1\}} \exp(\bar{\alpha}_k + \bar{\beta}_{k+1} + \bar{\gamma}_k)\right) \\ &- \ln\left(\sum_{\sim\{U_k=-1\}} \exp(\bar{\alpha}_k + \bar{\beta}_{k+1} + \bar{\gamma}_k)\right). \end{aligned} \tag{2.88}$$

An efficient calculation of this expression uses the following relation :

$$\begin{aligned} \ln(\exp(x) + \exp(y)) &= \max(x, y) + \ln(1 + \exp(-|x - y|)) \\ &\triangleq \overset{*}{\max}(x, y) \end{aligned} \tag{2.89}$$

which defines a modified-maximum function $\overset{*}{\max}$. The definition of this function extends to more than two arguments x and y, by proceeding recursively :

$$\begin{aligned} \ln(\exp(x) + \exp(y) + \exp(z)) &= \overset{*}{\max}(x, y, z) \\ &= \overset{*}{\max}(\overset{*}{\max}(x, y), z). \end{aligned} \tag{2.90}$$

Using this defintion, equation (2.88) becomes :

$$\begin{aligned} L_{U,k} &= \overset{*}{\max}_{\sim\{U_k=+1\}}(\bar{\alpha}_k + \bar{\beta}_{k+1} + \bar{\gamma}_k) \\ &- \overset{*}{\max}_{\sim\{U_k=-1\}}(\bar{\alpha}_k + \bar{\beta}_{k+1} + \bar{\gamma}_k) \end{aligned} \tag{2.91}$$

where $\overset{*}{\max}_{\sim\{U_k=u\}}$ denotes the modified maximization with respect to all variables

behalves U_k which is set to u.

The logarithmic expression of the α_k and β_k recursions (given by (2.28) and (2.29)) also use the modified maximum function. For $\bar{\alpha}_k$ this yields :

$$\bar{\alpha}_{k+1}(U_k, \ldots, U_{k-M+1}) = \overset{*}{\max_{U_{k-M}}}(\bar{\alpha}_k(U_{k-1}, \ldots, U_{k-M}) + \bar{\gamma}_k(X_k^1, \ldots, X_k^n)) \quad (2.92)$$

for coded bits X corresponding to information bits U. The initiation of this recursion becomes in the logarithmic domain :

$$\bar{\alpha}_1(\boldsymbol{T_1} = \mathbf{0}) = 0 \quad \text{and} \quad \bar{\alpha}_1(\boldsymbol{T_1} \neq \mathbf{0}) = -\infty. \quad (2.93)$$

Similarly the $\bar{\beta}_k$ recursion is :

$$\bar{\beta}_k(U_{k-1}, \ldots, U_{k-M}) = \overset{*}{\max_{U_k}}(\bar{\beta}_{k+1}(U_k, \ldots, U_{k-M+1}) + \bar{\gamma}_k(X_k^1, \ldots, X_k^n)) \quad (2.94)$$

with the following initialization, when the last encoder state is brought to zero :

$$\bar{\beta}_{K+1}(\boldsymbol{T_{K+1}} = \mathbf{0}) = 0 \quad \text{and} \quad \bar{\beta}_{K+1}(\boldsymbol{T_{K+1}} \neq \mathbf{0}) = -\infty, \quad (2.95)$$

otherwise the initialization consists in taking $\bar{\beta}_{K+1}$ uniform.

As for $\bar{\gamma}_k$, it has a simple expression in the case of binary-phase shift keying (BPSK) modulation and transmission over an AWGN channel. For example, for an AWGN channel with noise variance σ_n^2, BPSK modulation (thus $S_k^l = X_k^l$), and for the particular case of a rate $R = \frac{1}{2}$ code, the definition (2.27) becomes :

$$\begin{aligned} \bar{\gamma}_k(X_k^1, X_k^2) &= \ln\left(p(y_k^1|X_k^1) \cdot p(y_k^2|X_k^2)\right) \\ &= \ln\left(\frac{1}{\sigma_n\sqrt{2\pi}} \exp\left(-\frac{(y_k^1 - X_k^1)^2 + (y_k^2 - X_k^2)^2}{2\sigma_n^2}\right)\right) \\ &= -\ln(\sigma_n\sqrt{2\pi}) - \frac{(y_k^1 - X_k^1)^2 + (y_k^2 - X_k^2)^2}{2\sigma_n^2} \end{aligned} \quad (2.96)$$

Removing terms that do not depend on X_k^l, the previous expression of $\bar{\gamma}_k$ becomes :

$$\bar{\gamma}_k(X_k^1, X_k^2) = \frac{y_k^1 S_k^1 + y_k^2 S_k^2}{\sigma_n^2} \quad (2.97)$$

Note that using the $\max(\cdot)$ function above, instead of the modified-maximum function $\overset{*}{\max}(\cdot)$, gives the lower complexity MAX-LOG-MAP algorithm.

Relaying on orthogonal channels

3

The relay channel is a simple three-terminal network, where a relay helps a source terminal transmit to a destination. The particular case addressed in this chapter is the *orthogonal relay channel*, where the source terminal and the relay communicate on orthogonal channels.

After a description of the relay system model, this chapter presents three relaying protocols. These protocols are then briefly compared in terms of spectral efficiency on static channels and in terms of outage probability on fading channel.

Contents

3.1 INTRODUCTION

Users of a network can achieve higher transmission rates by relaying each others' signals. An elementary form of such cooperation is modeled by the relay channel, where a relay helps a pair of terminals communicate. The relay channel was introduced in [37], and its largest-known achievable rates were presented in [38].

The relay channel currently receives a lot of attention due to the perspectives cooperation offers in wireless networks : relaying provides resilience against shadowing, enhances the coverage of cellular networks, and provides so-called cooperative diversity.

Cooperative diversity consists in using relays to propagate redundant copies of a signal from a source to a destination. It enables to average fading of several point-to-point channels, much like spatial diversity in multiple-antennas transmissions. Fading and cooperative diversity both result from the broadcast nature of wireless transmission; fading comes from destructive interference between multiple propagation paths whereas cooperative diversity is achieved by exploiting redundancy between multiple network paths. Cooperative diversity was first explored in [39, 40, 41] and [42, 43, 44]. Subsequent analyzes closely related to the analysis of cooperative diversity include work on the diversity-multiplexing tradeoff [45] for the relay channel [46, 47, 48, 49]. Besides works on cooperative diversity, fundamental theoretical results for the wireless relay channel include most notably [50, 51].

This chapter describes the three categories of communication protocols generally used on the relay channel, and the associated achievable rates on Gaussian channels. The first protocol, called Decode-Forward (DF), consists in making the relay decode, re-encode and forward the signal. In the second protocol, Amplify-Forward (AF), the relay simply amplifies the signal it receives and forwards it to the destination. And finally, in the third protocol termed Compress-Forward (CF), the relay compresses the signal it receives and sends it to the destination.

Section 3.2 defines the model of the system under consideration. Then sections 3.3, 3.4 and 3.5 describe the three communication protocols and section 3.6 compares them.

3.2 SYSTEM MODEL

The relay channel consists of a source terminal, a relay terminal and a destination (fig. 3.1); in the case we consider, these terminals communicate wirelessly. Transmissions on wireless channels are broadcast by nature, thus the signal sent by the source to the destination is also heard by the relay. The role of the relay is, after the observation of this signal, to transmit any signal that improves the communication between the source and the destination.

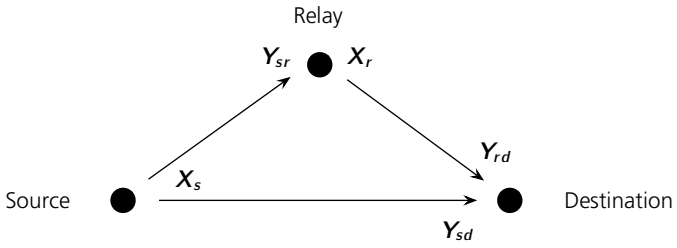


Figure 3.1: Relay channel.

The next two sections describe particular assumptions made in this dissertation and define the transmission models.

3.2.1 Medium Access

The relay acts both as a receiver and as a transmitter. However, current transceiver circuitry hardly permits reception and transmission at the same time in the same frequency band : the self-interference caused by the transmitted signal to the received signal can in general not be cancelled, because of the much larger power of the transmitted signal.

To account for this practical limitation, the relays we consider operate in half-duplex mode, i.e., they receive and transmit either at a different time, or in a different frequency band.

Furthermore, for half-duplex relays with separation in time only, the transmission of the relay and of the source occupy the same frequency band and interfere at the destination. To keep the destination receiver simple, we avoid such interference by making the source silent when the relay transmits.

Consequently, in our model, channels allocated to the source terminal and to the relay are completely orthogonal.

We call this the *orthogonal relay channel* model. By contrast, the *full-duplex relay channel* refers to the general case where the relay can transmit and receive at the same time in the same frequency band. The *half-duplex relay channel* refers to the case of a half-duplex relay that uses the same channel for the transmissions of the source terminal and of the relay.

3.2.2 Equivalent channel models

We model the transmissions by their discrete-time baseband equivalent model. The sequence $\boldsymbol{X_s}$ transmitted by the source is received by the relay and the destination as, respectively :

$$\boldsymbol{Y_{sr}} = H_{sr}\boldsymbol{X_s} + \boldsymbol{Z_{sr}} \tag{3.1}$$

$$\boldsymbol{Y_{sd}} = H_{sd}\boldsymbol{X_s} + \boldsymbol{Z_{sd}} \tag{3.2}$$

and is followed by the transmission of a sequence $\boldsymbol{X_r}$ from the relay to the destination, which receives

$$\boldsymbol{Y_{rd}} = H_{rd}\boldsymbol{X_r} + \boldsymbol{Z_{rd}}. \tag{3.3}$$

This chapter and the following, which address the theoretical characterization of the relay channel, use random complex Gaussian codewords $\boldsymbol{X_s}$ and $\boldsymbol{X_r}$. All elements of these codewords are mutually independent, identically distributed (i.i.d.) with zero mean and variances P_s and P_r respectively. We write the code rates R_s and R_r respectively. We assume that time and bandwidth are equally shared among the source terminal and the relay, thus the codewords have length $N/2$ for an overall transmission in N channel uses.

The additive noise terms $\boldsymbol{Z_{ij}}$ (with $i \in \{s, r\}$, $j \in \{r, d\}$) model the receiver noise and possibly other interferences affecting the system. Statistically, we model them as zero-mean, mutually independent, circularly symmetric complex Gaussian random variables. Their variance is normalized to $P_z = 1$, without loss of generality since the variances of the signals, P_s and P_r, can account for any signal-to-noise ratio.

The factors H_{ij} are the channel coefficients from terminal i to terminal j, with $i \in \{s, r\}$, $j \in \{r, d\}$. In this thesis, these factors capture the effects of path-loss and

frequency non-selective fading.

Path-loss refers to the attenuation of the propagating signal as the distance between transmitter and receiver increases. This attenuation affects the average power of the coefficients H_{ij}; a simple attenuation model is as follows :

$$\mathrm{E}\left\{|H_{ij}|^2\right\} = G/d^{\alpha} \tag{3.4}$$

where G is a constant depending on various parameters of the system, d is the distance between transmitter i and receiver j, and α is the path-loss exponent. According to field measurements, typical values of α lie between 2 and 6, depending on the environment [52]. The constant G is taken equal to 1 in this thesis, without loss of generality since several other degrees of freedom exist to account for different values.

Fading is due to reflection, refraction and diffusion of the signal on scattering objects in the environment. These objects can be buildings, vegetation, cars, or people, to name only a few. The propagation along multiple paths causes the receiver to get several copies of the signal. These copies have, in general, different attenuations, but also different phases and delays : they can thus add constructively or destructively at the receiver. The resulting variations of received signal strength are referred to as fading, and are modeled by the variations of H_{ij}. A simple model for fading considers that a large number of multipath components add at the receiver with independent random phases; in that case the coefficients H_{ij} can be modeled as circularly symmetric complex Gaussian random variables. This fading model is referred to as Rayleigh block fading because the amplitudes $|H_{ij}|$ follow a Rayleigh distribution and because, according to our model defined in (3.1)–(3.3), they are considered constant during the transmission of a block (i.e., N channel uses).

Finally, we also define the shorthand notations $\Gamma_{sj} = |H_{sj}|^2 P_s$ and $\Gamma_{rd} = |H_{rd}|^2 P_r$. Given the modelization above, the Γ_{ij} are exponential random variables with the following mean :

$$\sigma_{ij}^2 = \mathrm{E}\left\{|H_{ij}|^2\right\} P_i. \tag{3.5}$$

Their cumulative distribution function is

$$P(\Gamma_{ij} \leq \gamma_{ij}) = \begin{cases} 1 - e^{-\gamma_{ij}/\sigma_{ij}^2} & \text{for } \gamma_{ij} \geq 0, \\ 0 & \text{for } \gamma_{ij} < 0. \end{cases} \tag{3.6}$$

3.3 DECODE-FORWARD PROTOCOL

This sections presents the achievable spectral efficiencies with the DF protocol on the Gaussian orthogonal relay channel described in the previous section (the general case can be found in [38]). For the descriptions of the protocols we consider fixed realizations of the channel coefficients, known to all terminals; we write them h_{sd}, h_{sr} and h_{rd}. The corresponding fixed values of Γ_{sd}, Γ_{sr} and Γ_{rd} are written γ_{sd}, γ_{sr} and γ_{rd} respectively.

In the DF protocol, the relay decodes the codeword $\boldsymbol{X_s}$, and forwards a codeword $\boldsymbol{X_r}$ to the destination. The destination jointly decodes the codewords $\boldsymbol{X_s}$ and $\boldsymbol{X_r}$. Without going into the details here, the codes used by the source and the relay have equal code rates $R_r = R_s$. The spectral efficiency of the whole scheme is $R = R_r/2 = R_s/2$, where the factors $1/2$ account for using the channel twice in total, for each channel use by the source.

In DF, a first requirement for successful overall transmission is correct decoding at the relay. The constraint to satisfy is

$$\begin{aligned} R_s &< I(X_s; Y_{sr}) && (3.7)\\ &= \log_2(1+\gamma_{sr}). && (3.8) \end{aligned}$$

where the second equality results from the use of Gaussian codes on Gaussian channels[1].

The second constraint ensuring successful transmission regards the receiver at the destination. The destination is able to jointly decode the codewords $\boldsymbol{X_s}$ and $\boldsymbol{X_r}$ from the observation of $\boldsymbol{Y_{sd}}$ and $\boldsymbol{Y_{rd}}$ if and only if the overall spectral efficiency satisfies

$$\begin{aligned} R &\overset{(a)}{<} \frac{1}{2} I(X_s X_r; Y_{sd} Y_{rd}). && (3.9)\\ &\overset{(b)}{=} \frac{1}{2} I(X_s X_r; Y_{sd}|Y_{rd}) + \frac{1}{2} I(X_s X_r; Y_{rd})\\ &\overset{(c)}{=} \frac{1}{2} I(X_s; Y_{sd}) + \frac{1}{2} I(X_r; Y_{rd})\\ &\overset{(d)}{=} \frac{1}{2} \log_2(1+\gamma_{sd}) + \frac{1}{2} \log_2(1+\gamma_{rd}) && (3.10) \end{aligned}$$

where the factor $1/2$ in (a) comes from using the channel twice per symbol in X_s,

[1] Note that X_s and Y_{sr} are individual elements of the sequences $\boldsymbol{X_s}$ and $\boldsymbol{Y_{sr}}$. This notation convention is used for all sequences throughout this chapter.

where (b) follows from the chain rule for mutual information, (c) follows from the independence of X_r with respect to Y_{sd}, from the independence of Y_{rd} with respect to X_s and Y_{sd}, and from the independence of X_s with respect to Y_{rd}, and finally (d) follows from the use of Gaussian codes on Gaussian channels. Note that some authors use a slightly different expression that assumes repetition coding, as for example in [41].

The achievable spectral efficiencies with the DF protocol are thus :

$$\begin{cases} R & < & \frac{1}{2}\log_2(1+\gamma_{sr}), \\ R & < & \frac{1}{2}\log_2(1+\gamma_{sd}) + \frac{1}{2}\log_2(1+\gamma_{rd}). \end{cases} \tag{3.11}$$

3.4 AMPLIFY-FORWARD PROTOCOL

This section describes the achievable spectral efficiencies with the AF protocol of [41], again for fixed channels known to all terminals.

In the AF protocol, the relay simply amplifies the received signal $\boldsymbol{Y_{sr}}$ and forwards it to the destination. The forwarded signal is

$$\boldsymbol{X_r} = \sqrt{\frac{P_r}{|h_{sr}|^2 P_s + 1}}\,\boldsymbol{Y_{sr}} \tag{3.12}$$

$$= \sqrt{\frac{P_r}{|h_{sr}|^2 P_s + 1}}\,(h_{sr}\boldsymbol{X_s} + \boldsymbol{Z_{sr}}). \tag{3.13}$$

where the amplification factor ensures that the forwarded sequence has elements of variance P_r. The forwarded signal is received at the destination as

$$\begin{aligned} \boldsymbol{Y_{rd}} &= h_{rd}\boldsymbol{X_r} + Z_{rd} \\ &= h_{rd}\sqrt{\frac{P_r}{|h_{sr}|^2 P_s + 1}}\,h_{sr}\boldsymbol{X_s} + h_{rd}\sqrt{\frac{P_r}{|h_{sr}|^2 P_s + 1}}\,\boldsymbol{Z_{sr}} + \boldsymbol{Z_{rd}}. \end{aligned} \tag{3.14}$$

The destination first combines its received signals $\boldsymbol{Y_{rd}}$ and $\boldsymbol{Y_{rd}}$ with a maximum-ratio combiner, then decodes them.

Spectral efficiencies achievable with AF satisfy the constraint

$$R < \frac{1}{2}\,I(X_s; Y_{sd}Y_{rd}) \tag{3.15}$$

$$= \frac{1}{2}\,h(Y_{sd}Y_{rd}) - \frac{1}{2}\,h(Y_{sd}Y_{rd}|X_s) \tag{3.16}$$

where $h()$ denotes the differential entropy. Moreover, since Y_{sd}, Y_{rd} and X_s are correlated Gaussian random variables, these differential entropies are [53] :

$$h(Y_{sd}Y_{rd}) = \log_2\left((2\pi e)^2|\boldsymbol{Q}_{\boldsymbol{Y_{sd}Y_{rd}}}|\right) \tag{3.17}$$

$$h(Y_{sd}Y_{rd}|X_s) = \log_2\left((2\pi e)^2\frac{|\boldsymbol{Q}_{\boldsymbol{X_sY_{sd}Y_{rd}}}|}{|\boldsymbol{Q}_{\boldsymbol{X_s}}|}\right), \tag{3.18}$$

where $\boldsymbol{Q}_{\boldsymbol{Y_{sd}Y_{rd}}}$, $\boldsymbol{Q}_{\boldsymbol{X_sY_{sd}Y_{rd}}}$ and $\boldsymbol{Q}_{\boldsymbol{X_s}}$ are covariance matrices defined below. The covariance matrix of Y_{sd} and Y_{rd} is

$$\begin{aligned}\boldsymbol{Q}_{\boldsymbol{Y_{sd}Y_{rd}}} &\triangleq \mathrm{E}\left\{\begin{bmatrix} Y_{sd} \\ Y_{rd}\end{bmatrix}\begin{bmatrix} Y_{sd} & Y_{rd}\end{bmatrix}^*\right\} \\ &= \begin{bmatrix} |h_{sd}|^2P_s+1 & h_{sd}h_{sr}^*h_{rd}^*P_s\sqrt{\frac{P_r}{|h_{sr}|^2P_s+1}} \\ h_{sd}^*h_{sr}h_{rd}P_s\sqrt{\frac{P_r}{|h_{sr}|^2P_s+1}} & |h_{rd}|^2P_r+1\end{bmatrix}\end{aligned} \tag{3.19}$$

and its determinant is

$$\begin{aligned}|\boldsymbol{Q}_{\boldsymbol{Y_{sd}Y_{rd}}}| &= 1+|h_{sd}|^2P_s+|h_{rd}|^2P_r+\frac{|h_{sd}|^2|h_{rd}|^2P_sP_r}{|h_{sr}|^2P_s+1} \\ &= 1+\gamma_{sd}+\gamma_{rd}+\frac{\gamma_{sd}\gamma_{rd}}{\gamma_{sr}+1}.\end{aligned} \tag{3.20}$$

The covariance matrix of X_s, Y_{sd} and Y_{rd} is

$$\boldsymbol{Q}_{\boldsymbol{X_sY_{sd}Y_{rd}}} \triangleq \mathrm{E}\left\{\begin{bmatrix} X_s \\ Y_{sd} \\ Y_{rd}\end{bmatrix}\begin{bmatrix} X_s & Y_{sd} & Y_{rd}\end{bmatrix}^*\right\} \tag{3.21}$$

$$= \begin{bmatrix} P_s & h_{sd}^*P_s & h_{sr}^*h_{rd}^*P_s\sqrt{\frac{P_r}{|h_{sr}|^2P_s+1}} \\ h_{sd}P_s & |h_{sd}|^2P_s+1 & h_{sd}h_{sr}^*h_{rd}^*P_s\sqrt{\frac{P_r}{|h_{sr}|^2P_s+1}} \\ h_{sr}h_{rd}P_s\sqrt{\frac{P_r}{|h_{sr}|^2P_s+1}} & h_{sd}^*h_{sr}h_{rd}P_s\sqrt{\frac{P_r}{|h_{sr}|^2P_s+1}} & |h_{rd}|^2P_r+1\end{bmatrix}$$

and has a determinant equal to

$$\begin{aligned}|\boldsymbol{Q}_{\boldsymbol{X_sY_{sd}Y_{rd}}}| &= P_s+\frac{|h_{rd}|^2P_rP_s}{|h_{sr}|^2P_s+1} \\ &= P_s\left(1+\frac{\gamma_{rd}}{\gamma_{sr}+1}\right).\end{aligned} \tag{3.22}$$

Finally, we obviously have

$$\begin{aligned} \boldsymbol{Q}_{\boldsymbol{X}_s} &= P_s & (3.23)\\ &= |\boldsymbol{Q}_{\boldsymbol{X}_s}|. & (3.24) \end{aligned}$$

The achievable spectral efficiencies defined in (3.16) finally become, using (3.17)–(3.24) :

$$R < \frac{1}{2}\log_2\left(1+\gamma_{sd}+\frac{\gamma_{sr}\gamma_{rd}}{1+\gamma_{sr}+\gamma_{rd}}\right). \quad (3.25)$$

3.5 COMPRESS-FORWARD PROTOCOL

A relay using the AF protocol forwards the signal $\boldsymbol{Y_{sr}}$ in an analog way; in contrast, a relay using the CF protocol forwards a digital representation of the signal $\boldsymbol{Y_{sr}}$ it receives.

The number of bits for representing and forwarding the signal $\boldsymbol{Y_{sr}}$ is limited by the rate R_r available on the relay-to-destination channel. In general thus, only a compressed representation $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$ of $\boldsymbol{Y_{sr}}$ can be forwarded; moreover, lossy compression might have to be used to meet the rate constraint. This is obviously always the case for continuous channels.

A simple method for compressing $\boldsymbol{Y_{sr}}$ is to use rate-distortion source coding [54]. But in the CF protocol, compression can be improved by exploiting the knowledge of the signal $\boldsymbol{Y_{sd}}$ at the destination, because this signal is correlated with $\boldsymbol{Y_{sr}}$. The CF protocol as described in [38] uses such a compression method, referred to as compression with side-information, or Wyner-Ziv coding [55].

The next section presents rate-distortion source coding and Wyner-Ziv coding; in the following one comes the description of the rates achievable with the CF protocol.

3.5.1 Rate-distortion coding and Wyner-Ziv source coding

Rate-distortion coding The goal of rate-distortion coding in this context is to compress, or quantize, the symbols sequence $\boldsymbol{Y_{sr}}$ of length $N/2$ into a binary representation of $R_rN/2$ bits. To do so, the source encoder uses a codebook of $2^{R_rN/2}$ codewords $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}(W)$, indexed by $W \in \{1,\ldots,2^{R_rN/2}\}$. The source encoder maps the sequence $\boldsymbol{Y_{sr}}$ to a codeword $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}(W)$ and outputs the index W (see fig. 3.2).

In the case of a continuous variable $\boldsymbol{Y_{sr}}$, the reconstruction $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}(W)$ is inevitably a

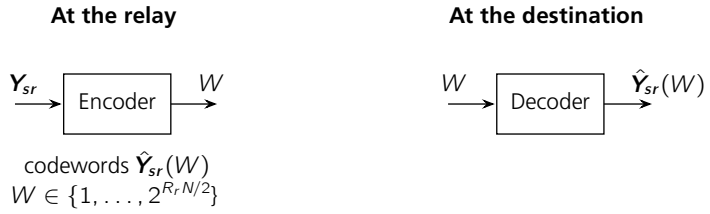


Figure 3.2: Rate-distortion coding in the context of the relay channel.

distorted representation of $\boldsymbol{Y_{sr}}$. In the rate-distortion problem, the distortion measure between the two sequences is the average of per-symbol distortions. That is, for a per-symbol distortion measure $d(Y_{sr,i}, \hat{Y}_{sr,i})$ ($i = 1, \ldots, N/2$), the average distortion between $\boldsymbol{Y_{sr}}$ and $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}(W)$ is defined as

$$\frac{2}{N} \mathrm{E} \left\{ \sum_{i=1}^{N/2} d(Y_{sr,i}, \hat{Y}_{sr,i}) \right\}. \tag{3.26}$$

A given average distortion level D can only be achieved if the quantization codebook contains a sufficient number of (well-chosen) codewords $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$. Rate-distortion theory determines how to generate such a codebook; in particular it shows that the constraint on the rate R_r, and thus on the number of codewords required in the codebook, is the following :

$$R_r(D) > \min_{P_{\hat{Y}_{sr}|Y_{sr}} : \mathrm{E}\{d(Y_{sr}, \hat{Y}_{sr})\} < D} I(Y_{sr}; \hat{Y}_{sr}). \tag{3.27}$$

Thus, with a well-chosen codebook of at least $2^{R_r(D)N/2}$ codewords it is always possible to find a codeword within distortion D of the sequence $\boldsymbol{Y_{sr}}$.

Wyner-Ziv coding In the CF protocol, a lower distortion D can be achieved for a same rate R_r, by taking advantage of the statistical dependence between $\boldsymbol{Y_{sr}}$ and the signal $\boldsymbol{Y_{sd}}$ known at the destination. The sequence $\boldsymbol{Y_{sd}}$ is then referred to as *side-information*; it is available at the source decoder (destination) but not at the source encoder (relay) (see fig. 3.3). Such coding with side-information at the receiver is termed Wyner-Ziv coding (see [55]).

The source encoder uses a codebook of codewords $\boldsymbol{U}(W, V)$ indexed by W and V. It maps the sequence $\boldsymbol{Y_{sr}}$ to one of these codewords and outputs the index $W \in$

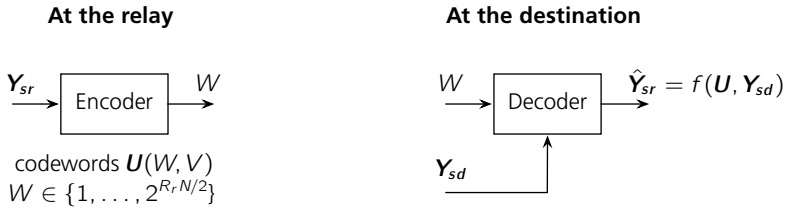


Figure 3.3: Wyner-Ziv coding in the context of the relay channel.

$\{1,\ldots,2^{R_r N/2}\}$. The source decoder receives this index W and the side-information $\boldsymbol{Y_{sd}}$ correlated with $\boldsymbol{Y_{sr}}$. Using these two pieces of information, the decoder finds the missing index V and thus the codeword $\boldsymbol{U}(W,V)$. Finally, the estimation of $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$ is a function of both the codeword $\boldsymbol{U}(W,V)$ and the side-information $\boldsymbol{Y_{sd}}$, i.e.,

$$\hat{Y}_{sr,i} = f(U_i, Y_{sd,i}) \tag{3.28}$$

for $i = 1,\ldots,N/2$.

Achieving a distortion D requires the following rate :

$$R_r(D) > \min_{P_{U|Y_{sr}}, f(.)\,:\,\mathrm{E}\{d(Y_{sr},f(U,Y))\}<D} I(Y_{sr};U) - I(Y_{sd};U). \tag{3.29}$$

Note that the minimization is over $P_{U|Y_{sr}}$, which defines the statistical relation between U and Y_{sr} (and thus the way the codewords $\boldsymbol{U}$ are generated), and over the estimation function $f(\cdot)$.

The mutual informations in (3.29) can be rewritten as follows :

$$\begin{aligned} I(Y_{sr};U) - I(Y_{sd};U) &= H(U|Y_{sd}) - H(U|Y_{sr}) \\ &\overset{(a)}{=} H(U|Y_{sd}) - H(U|Y_{sr}Y_{sd}) \\ &= I(Y_{sr};U|Y_{sd}) \\ &\overset{(b)}{=} I(Y_{sr};U\,\hat{Y}_{sr}|Y_{sd}) \\ &\overset{(c)}{=} I(Y_{sr};\hat{Y}_{sr}|Y_{sd}) + I(Y_{sr};U|\hat{Y}_{sr}Y_{sd}) \\ &\geq I(Y_{sr};\hat{Y}_{sr}|Y_{sd}), \end{aligned} \tag{3.30}$$

where (a) follows from the fact that $U - Y_{sr} - Y_{sd}$ is a Markov chain (this depends on the code construction, for more details see [55]); (b) follows from the fact that $\hat{Y}_{sr}$ is a

function of U and Y_{sd}; and (c) follows from the chain rule for mutual information. The inequality above becomes an equality if $I(Y_{sr}; U|\hat{Y}_{sr}Y_{sd}) = 0$. This happens in the CF protocol, because the function $f(\cdot)$ is chosen such that

$$\hat{Y}_{sr} = f(U, Y_{sd}) = U.$$

Thus, the achievable rates for Wyner-Ziv coding in the CF protocol are

$$R_r(D) > \min_{P_{\hat{Y}_{sr}|Y_{sr}} : \mathrm{E}\left\{d(Y_{sr},\hat{Y}_{sr})\right\}<D} I(Y_{sr}; \hat{Y}_{sr}|Y_{sd}). \tag{3.31}$$

Observe that if the source encoder also knew the side information $\boldsymbol{Y_{sd}}$, the achievable rates would be :

$$R_r(D) > \min_{P_{\hat{Y}_{sr}|Y_{sr}Y_{sd}} : \mathrm{E}\{d(Y_{sr},Y_{sr})\}<D} I(Y_{sr}; \hat{Y}_{sr}|Y_{sd}),$$

this rate being different from (3.31) in that $\hat{Y}_{sr}$ is allowed to have a statistical dependence with Y_{sd} described by $P_{\hat{Y}_{sr}|Y_{sr}Y_{sd}}$.

3.5.2 Achievable spectral efficiencies with CF

In the CF protocol, the relay observes $\boldsymbol{Y_{sr}}$ and produces a corresponding compression index W. This index W is sent with a codeword $\boldsymbol{X_r}$, received as $\boldsymbol{Y_{rd}}$ by the destination. In order to decode $\boldsymbol{X_s}$ from $\boldsymbol{Y_{sd}}$ and $\boldsymbol{Y_{rd}}$, the destination performs the following operations :

- First, the destination decodes $\boldsymbol{X_r}$ (and thus W) from $\boldsymbol{Y_{rd}}$;
- Second, using the index W along with the side information $\boldsymbol{Y_{sd}}$, it computes estimates $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$ of $\boldsymbol{Y_{sr}}$;
- And third, it decodes $\boldsymbol{X_s}$ from $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$ and $\boldsymbol{Y_{sd}}$.

These decoding steps are successful for appropriate choices of the code parameters; the relevant constraints on these parameters are detailed in the following.

Decoding $\boldsymbol{X_r}$ Successfully decoding $\boldsymbol{X_r}$ from $\boldsymbol{Y_{rd}}$ is possible for code rates R_r satisfying the following constraint :

$$\begin{aligned} R_r &< I(X_r; Y_{rd}) && (3.32)\\ &= \log_2(1+\gamma_{rd}). && (3.33) \end{aligned}$$

where the second equality follows from the use of Gaussian codewords on a Gaussian channel.

Decompression (Wyner-Ziv decoding) From W and $\boldsymbol{Y_{sd}}$, the receiver tries to obtain $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$. With our Gaussian assumptions on the channels and on their inputs, and with compression of $\boldsymbol{Y_{sr}}$ so as to minimize the mean square error on the estimates $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$, these can be modeled as

$$\hat{\boldsymbol{Y}}_{\boldsymbol{sr}} = \boldsymbol{Y_{sr}} + \boldsymbol{Z_q}, \tag{3.34}$$

where $\boldsymbol{Z_q}$ is a sequence of i.i.d. circularly symmetric complex Gaussian quantization noise samples of variance N_q. Note that minimizing the mean square error on the estimates $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$ does not necessarily maximize the overall achievable rate R; this choice is made by default since the optimal distribution for $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$ is not known.

The achievable level of noise variance N_q depends on the code rate used for transmission on the relay-to-destination channel. The constraint is as follows :

$$\begin{aligned} R_r &> I(Y_{sr}; \hat{Y}_{sr}|Y_{sd}) && (3.35)\\ &= h(Y_{sr}|Y_{sd}) - h(Y_{sr}|\hat{Y}_{sr}Y_{sd}) \\ &= \log_2\left(2\pi e\frac{|\boldsymbol{Q}_{\boldsymbol{Y_{sr}Y_{sd}}}|}{|\boldsymbol{Q}_{\boldsymbol{Y_{sd}}}|}\right) - \log_2\left(2\pi e\frac{|\boldsymbol{Q}_{\boldsymbol{Y_{sr}\hat{Y}_{sr}Y_{sd}}}|}{|\boldsymbol{Q}_{\boldsymbol{\hat{Y}_{sr}Y_{sd}}}|}\right) \end{aligned}$$

where the last equality follows from the use of Gaussian codewords and channels.

The determinants of the covariance matrices in (3.35) are as follows. The covariance

matrix of Y_{sr}, $\hat{Y}_{sr}$ and Y_{sd} is

$$\begin{aligned} \boldsymbol{Q}_{\boldsymbol{Y}_{sr}\hat{\boldsymbol{Y}}_{sr}\boldsymbol{Y}_{sd}} &\triangleq \mathrm{E}\left\{ \begin{bmatrix} Y_{sr} \\ \hat{Y}_{sr} \\ Y_{sd} \end{bmatrix} \begin{bmatrix} Y_{sr} & \hat{Y}_{sr} & Y_{sd} \end{bmatrix}^* \right\} \\ &= \begin{bmatrix} |h_{sr}|^2P_s+1 & |h_{sr}|^2P_s+1 & h_{sr}h_{sd}^*P_s \\ |h_{sr}|^2P_s+1 & |h_{sr}|^2P_s+1+N_q & h_{sr}h_{sd}^*P_s \\ h_{sr}^*h_{sd}P_s & h_{sr}^*h_{sd}P_s & |h_{sr}|^2P_s+1 \end{bmatrix} \end{aligned} \tag{3.36}$$

and has a determinant equal to

$$\begin{aligned} |\boldsymbol{Q}_{\boldsymbol{Y}_{sr}\hat{\boldsymbol{Y}}_{sr}\boldsymbol{Y}_{sd}}| &= N_q\,(1+|h_{sr}|^2P_s+|h_{sd}|^2P_s) \\ &= N_q\,(1+\gamma_{sr}+\gamma_{sd}). \end{aligned} \tag{3.37}$$

The covariance matrix between Y_{sr} and Y_{sd} is

$$\begin{aligned} \boldsymbol{Q}_{\boldsymbol{Y}_{sr}\boldsymbol{Y}_{sd}} &\triangleq \mathrm{E}\left\{ \begin{bmatrix} Y_{sr} \\ Y_{sd} \end{bmatrix} \begin{bmatrix} Y_{sr} & Y_{sd} \end{bmatrix}^* \right\} \\ &= \begin{bmatrix} |h_{sr}|^2P_s+1 & h_{sr}h_{sd}^*P_s \\ h_{sr}^*h_{sd}P_s & |h_{sr}|^2P_s+1 \end{bmatrix} \end{aligned} \tag{3.38}$$

and its determinant is

$$\begin{aligned} |\boldsymbol{Q}_{\boldsymbol{Y}_{sr}\boldsymbol{Y}_{sd}}| &= 1+|h_{sr}|^2P_s+|h_{sd}|^2P_s \\ &= 1+\gamma_{sr}+\gamma_{sd}. \end{aligned} \tag{3.39}$$

Finally, the covariance matrix of $\hat{Y}_{sr}$ and Y_{sd} is

$$\begin{aligned} \boldsymbol{Q}_{\hat{\boldsymbol{Y}}_{sr}\boldsymbol{Y}_{sd}} &\triangleq \mathrm{E}\left\{ \begin{bmatrix} \hat{Y}_{sr} \\ Y_{sd} \end{bmatrix} \begin{bmatrix} \hat{Y}_{sr} & Y_{sd} \end{bmatrix}^* \right\} \\ &= \begin{bmatrix} |h_{sr}|^2P_s+1+N_q & h_{sr}h_{sd}^*P_s \\ h_{sr}^*h_{sd}P_s & |h_{sr}|^2P_s+1 \end{bmatrix} \end{aligned} \tag{3.40}$$

and has a determinant equal to

$$\begin{aligned} |\boldsymbol{Q}_{\hat{\boldsymbol{Y}}_{sr}\boldsymbol{Y}_{sd}}| &= (1+N_q)(1+|h_{sd}|^2P_s)+|h_{sr}|^2P_s \\ &= (1+N_q)(1+\gamma_{sd})+\gamma_{sr}. \end{aligned} \tag{3.41}$$

Substituting these determinants in (3.35) gives

$$R_r > \log_2\left(1 + \frac{1 + \frac{\gamma_{sr}}{1+\gamma_{sd}}}{N_q}\right). \tag{3.42}$$

This specifies the rate R_r needed to have estimates $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$ of $\boldsymbol{Y}_{\boldsymbol{sr}}$ with compression noise level N_q. If conventional rate-distortion source coding is used instead of Wyner-Ziv coding, i.e., when the side information is not taken into account, the rate R_r and the compression noise variance N_q are related through the constraint

$$R_r > \log_2\left(1 + \frac{1+\gamma_{sr}}{N_q}\right) \tag{3.43}$$

which is the same as above with $\gamma_{sd} = 0$.

Decoding $\boldsymbol{X_s}$ Once the relay has decoded $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$, it can decode $\boldsymbol{X_s}$ if the overall transmission rate satisfies

$$\begin{aligned} R &< \frac{1}{2} I(X_s; \hat{Y}_{sr} Y_{sd}) \qquad (3.44)\\ &= \frac{1}{2} h(\hat{Y}_{sr} Y_{sd}) - \frac{1}{2} h(\hat{Y}_{sr}|X_s) - \frac{1}{2} h(Y_{sd}|X_s) \\ &= \frac{1}{2}\log_2\left((2\pi e)^2 |\boldsymbol{Q}_{\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}\boldsymbol{Y}_{\boldsymbol{sd}}}|\right) - \frac{1}{2}\log_2\left(2\pi e |\boldsymbol{Q}_{\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}|\boldsymbol{X}_{\boldsymbol{s}}}|\right) - \frac{1}{2}\log_2\left(2\pi e |\boldsymbol{Q}_{\boldsymbol{Y}_{\boldsymbol{sd}}|\boldsymbol{X}_{\boldsymbol{s}}}|\right) \end{aligned}$$

where the factor $1/2$ comes from using the channel twice per symbol X_s and where the last equality comes from the use of Gaussian codes and channels. With $|\boldsymbol{Q}_{\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}\boldsymbol{Y}_{\boldsymbol{sd}}}|$ given by (3.41), and since $\hat{Y}_{sr}|X_s$ and $Y_{sd}|X_s$ are Gaussian distributed with variance $1 + N_q$ and 1 respectively, the constraint (3.44) becomes

$$R < \frac{1}{2}\log_2\left(1 + \gamma_{sd} + \frac{\gamma_{sr}}{1+N_q}\right). \tag{3.45}$$

Summary : achievable spectral efficiencies To sum up the results above, the spectral efficiencies achievable with CF have to satisfy the following three constraints :

$$\begin{cases} R &< \frac{1}{2}\log_2\left(1 + \gamma_{sd} + \frac{\gamma_{sr}}{1+N_q}\right) \\ R_r &< \log_2(1+\gamma_{rd}) \\ R_r &> \log_2\left(1 + \frac{1+\frac{\gamma_{sr}}{1+\gamma_{sd}}}{N_q}\right). \end{cases} \tag{3.46}$$

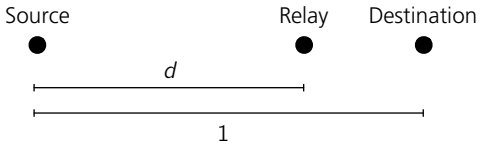


Figure 3.4: Geometric configuration for the numerical evaluations of this section.

3.6 COMPARISON OF THE PROTOCOLS

This section compares the three cooperative protocols in two different situations :

- First a comparison of spectral efficiencies for static channels;
- Second a comparison of outage on fading channels.

The comparisons are made for the setup of fig. 3.4 : the source terminal and the destination are at a normalized distance 1, and the relay is on the line joining them at a distance d from the source terminal.

Static channels In this case, the channel coefficients h_{ij} include the effects of path-loss but not those of fading. The path-loss exponent is set equal to $\alpha = 3.5$. We have

$$\begin{aligned} |h_{sr}|^2 &= 1/d^{\alpha} \\ |h_{rd}|^2 &= 1/(1-d)^{\alpha} \\ |h_{sd}|^2 &= 1. \end{aligned}$$

The three cooperative protocols are also compared with non-cooperative transmission, i.e., transmission by the source only during the two time slots. The corresponding achievable spectral efficiencies are of course

$$R < \log_2(1+\gamma_{sd}). \tag{3.47}$$

The largest achievable spectral efficiencies are plotted in fig. 3.5, for transmission powers set to $P_s = P_r = 3$. The highest spectral efficiencies are achieved by the DF protocol when the relay is close to the source terminal, and by non-cooperative transmission when the relay is close to the destination. Note that for a full-duplex relay

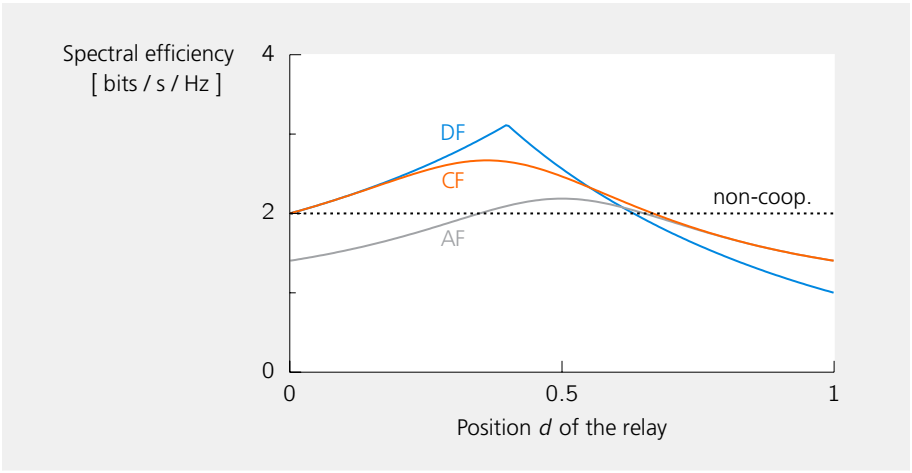


Figure 3.5: Spectral efficiencies achieved with the DF, AF and CF protocols, as well as with non-cooperative transmission.

channel, all three cooperative protocols would achieve higher spectral efficiencies than non-cooperative transmission.

Fading channels In the case of fading channels, we generate channel coefficients H_{ij} that include both the effects of path-loss and those of fading. We thus have

$$\begin{aligned} \sigma_{sr}^2 &= P_s/d^\alpha \\ \sigma_{rd}^2 &= P_r/(1-d)^\alpha \\ \sigma_{sd}^2 &= P_s. \end{aligned}$$

with again a path-loss exponent $\alpha = 3.5$. The source terminal attempts a transmission at a spectral efficiency $R = 1$; an outage occurs when $R = 1$ is not achievable for the realizations of the channel coefficients. The transmission powers are set to $P_s = P_r = 10$.

Channel state information (CSI) is assumed at all receivers in all protocols (each receiver knows the channel coefficient h_{ij} for the signals it receives). In the AF case, the receiver at the destination also needs to know h_{sr} to perform maximum ratio combining. In the CF case, the transmitter at the relay needs to know either h_{rd} or h_{sd} in order to choose its encoding parameters N_q and R_r.

As seen in fig. 3.6, non-cooperative transmission is worse than all cooperative proto-

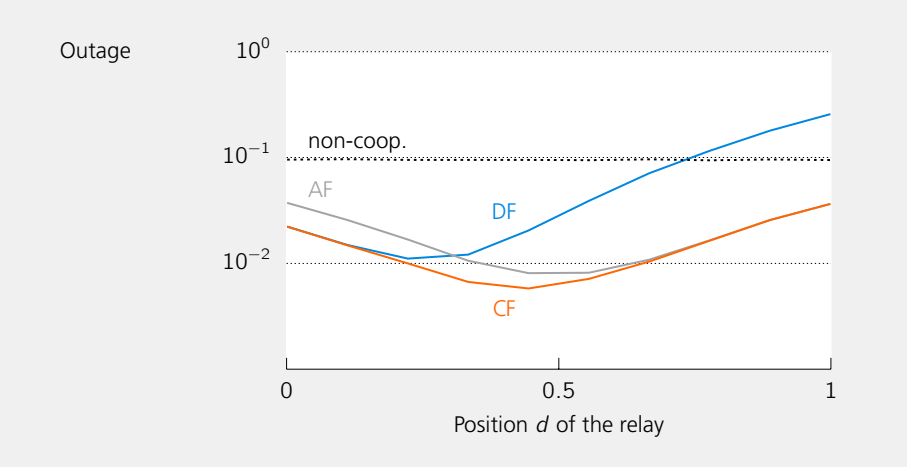


Figure 3.6: Outage achieved with the DF, AF and CF protocols, as well as with non-cooperative transmission.

cols when the relay is close to the source. The DF protocol performs better than the AF protocol when the relay is close to the source, otherwise AF performs better. The CF protocol achieves the lowest outage, but the comparison with the other protocols is biased because it uses additional CSI at the relay, and can thus partially adapt to the channel.

In contrast, the next chapter considers the CF protocol without additional CSI at the relay, and discusses the optimal choice of the parameters N_q and R_r.

Adaptive Compress-Forward relaying in fading environments

4

This chapter addresses the Compress-Forward (CF) protocol over a wireless relay network with orthogonal transmissions from the source and the relay terminals. We show that when the transmitters have no instantaneous channel state information the optimal compression parameters often make Wyner-Ziv coding reduce to usual rate-distortion coding, i.e. compression that does not take into account the side information available at the destination.

This result simplifies the implementation of the CF protocol in the case we consider, since it shows that in several situations one can use more convenient compression methods without significant performance loss.

However, not being to use the side information efficiently makes the CF protocol achieve a higher outage than the Amplify-Forward (AF) protocol. In contrast, we highlight that CF becomes better than AF when the relay knows the level of side information available at the destination.

These results are an original contribution of this thesis; they were first reported in [3, 4].

Contents

4.1 INTRODUCTION

This chapter considers a hybrid protocol using DF when the relay can decode successfully, and CF otherwise. The focus is on the CF part of the protocol.

In the CF technique, Wyner-Ziv coding is used by the relay because the destination uses the signal received directly from the source as side information [38, 55]. A practical CF strategy with Wyner-Ziv coding is presented in [56], for fixed channels known to all terminals. However other contributions as [57] have simplified the protocol by using, instead of Wyner-Ziv coding, usual rate-distortion coding that does not take into account the side information available at the destination.

In this work we investigate Wyner-Ziv coding in the case of fading channels unknown to the transmitters and observe that, under some conditions, the compression parameters minimizing outage make Wyner-Ziv coding reduce to usual rate-distortion coding. This observation has some importance because rate-distortion coding is simpler than Wyner-Ziv coding, because many CF schemes developed in the literature have adopted rate-distortion coding, and because situations for which rate-distortion coding is optimal can be rather frequent.

As such however the CF protocol yields a higher outage than the AF protocol; this contrasts with the case of static channels, on which CF compares favorably to AF in terms of transmission rate. To lower the outage of CF, one can inform the relay of the side information available at the destination. This makes CF achieve a lower outage than AF, even though only slightly so.

The chapter is organized as follows. The system model, the protocol used and the relevant information theoretic relations are presented in section 4.2. Section 4.3 derives the optimal compression parameters and section 4.4 presents an upper bound on the probability that the optimal compression method would effectively be Wyner-Ziv. Section 4.5 illustrates the results and section 4.6 compares CF with AF; finally section 4.7 concludes the chapter.

4.2 SYSTEM MODEL

We consider a simple wireless relay network consisting of a source terminal, a relay and a destination as in fig. 4.1. Transmissions suffer from frequency-flat block-fading and additive noise, which is deemed appropriate for a narrow-band low-mobility scenario.

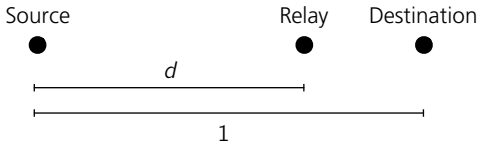


Figure 4.1: Relay network and geometric configuration.

4.2.1 Medium Access

The relay operates in half-duplex mode, to get round the technical difficulty of receiving and transmitting at the same time in the same frequency band. Moreover, in order to keep the receiver at the destination simple, we allocate orthogonal channels to the source and the relay. Bandwidth and time are equally shared out among these.

Instantaneous channel state information (CSI) is available at the receivers, whereas the transmitters can obtain and use average CSI only.

4.2.2 Transmission model

We model the transmissions by their discrete-time baseband equivalent models. The sequence $\boldsymbol{X_s}$ transmitted by the source is received by the relay and the destination as, respectively :

$$\boldsymbol{Y_{sr}} = H_{sr}\boldsymbol{X_s} + \boldsymbol{Z_{sr}} \tag{4.1}$$

$$\boldsymbol{Y_{sd}} = H_{sd}\boldsymbol{X_s} + \boldsymbol{Z_{sd}} \tag{4.2}$$

and is followed by the transmission of a sequence $\boldsymbol{X_r}$ from the relay to the destination, which receives

$$\boldsymbol{Y_{rd}} = H_{rd}\boldsymbol{X_r} + \boldsymbol{Z_{rd}}. \tag{4.3}$$

The input sequences $\boldsymbol{X_s}$ and $\boldsymbol{X_r}$ are codewords of length $N/2$. We choose to use a random code with complex Gaussian symbols, thus all elements of these sequences are mutually independent, identically distributed (i.i.d.) with zero mean and variances P_s and P_r respectively. We write the code rates R_s and R_r respectively. The factors H_{ij} are the channel fading coefficients from terminal i to terminal j and the $\boldsymbol{Z_{ij}}$ are sequences of additive noise, with $i \in \{s, r\}, j \in \{r, d\}$. Statistically, we model the H_{ij} and $\boldsymbol{Z_{ij}}$ as zero-mean, mutually independent, circularly symmetric complex Gaussian random variables. The noise samples in $\boldsymbol{Z_{ij}}$ have a variance normalized to $N_{ij} = 1$.

Finally, we also define the shorthand notations $\Gamma_{sj} = |H_{sj}|^2 P_s$ and $\Gamma_{rd} = |H_{rd}|^2 P_r$. Given the modelization above, these are exponential random variables with mean $\sigma_{ij}^2 \triangleq \mathrm{E}\left\{|H_{ij}|^2\right\} P_i$.

4.2.3 Cooperation protocol

Our cooperation protocol extends over two slots; the source transmits in the first slot and the relay forwards in the second one. The processing at the relay depends on the quality of the signal received from the source. If the relay is able to decode the signal, it forwards it using DF, otherwise it uses CF. The reason for using this hybrid DF-CF strategy is to focus our analysis of the CF protocol only on situations in which DF is not possible.

Achievable rates with these protocols were derived in [38], and particularized to the Gaussian orthogonal relay channel in chapter 3. Main results are summarized below.

DF relaying

If the relay is able to decode $\boldsymbol{X_s}$ from $\boldsymbol{Y_{sr}}$, that is when

$$R_s < \log_2(1 + \Gamma_{sr}), \tag{4.4}$$

then the DF protocol is used and the sequence $\boldsymbol{X_r}$ is taken from a code with rate $R_r = R_s$ and length $N/2$. The destination jointly decodes the signals from the source and the relay. Achievable spectral efficiencies R for the whole cooperative transmission are those satisfying the constraint

$$R < \frac{1}{2}\log_2(1 + \Gamma_{sd}) + \frac{1}{2}\log_2(1 + \Gamma_{rd}). \tag{4.5}$$

Note that we obviously have $R_s = R_r = 2R$.

CF relaying

When the relay cannot decode the signal from the source it uses the CF protocol. In this protocol the relay performs Wyner-Ziv coding, i.e., it compresses the sequence $\boldsymbol{Y_{sr}}$ and sends a corresponding codeword $\boldsymbol{X_r}$; then, from these bits $\boldsymbol{X_r}$ and from $\boldsymbol{Y_{sd}}$ considered as side information, the destination calculates estimates $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$ of $\boldsymbol{Y_{sr}}$. Finally, the destination decodes $\boldsymbol{X_s}$ from $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$ and $\boldsymbol{Y_{sd}}$.

In the following, we first characterize the Wyner-Ziv estimates $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$, and then give the rate constraints to be satisfied for successful decoding.

The level of compression of $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$ is controlled by two parameters, namely the code rate R_r chosen for forwarding on the relay-to-destination channel and the quantization noise variance N_q. With our Gaussian assumptions on the channels and their inputs, and with compression of $\boldsymbol{Y}_{\boldsymbol{sr}}$ so as to minimize the mean square error on the estimates $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$, these can be modeled as

$$\hat{\boldsymbol{Y}}_{\boldsymbol{sr}} = \boldsymbol{Y}_{\boldsymbol{sr}} + \boldsymbol{Z}_{\boldsymbol{q}}, \tag{4.6}$$

where $\boldsymbol{Z}_{\boldsymbol{q}}$ is a sequence of i.i.d. circularly symmetric complex Gaussian quantization noise samples of variance N_q.

At the destination, the receiver first considers $\boldsymbol{Y}_{\boldsymbol{rd}}$ and tries to decode $\boldsymbol{X}_{\boldsymbol{r}}$. The event $\mathcal{C}$ of *correct decoding* is given by

$$\mathcal{C}: \quad R_r < \log_2(1+\Gamma_{rd}). \tag{4.7}$$

Then, from $\boldsymbol{X}_{\boldsymbol{r}}$ and $\boldsymbol{Y}_{\boldsymbol{sd}}$ the receivers tries to obtain $\hat{\boldsymbol{Y}}_{\boldsymbol{sr}}$; the event $\mathcal{D}$ of *correct decompression* is written

$$\mathcal{D}: \quad R_r > \log_2\left(1+\frac{1+\frac{\Gamma_{sr}}{1+\Gamma_{sd}}}{N_q}\right). \tag{4.8}$$

If both $\mathcal{C}$ and $\mathcal{D}$ occur, the achievable spectral efficiencies for the cooperative transmission are those satisfying

$$R < \frac{1}{2}\log_2\left(1+\Gamma_{sd}+\frac{\Gamma_{sr}}{1+N_q}\right) \triangleq I_{CF}. \tag{4.9}$$

Otherwise the signal from the relay is unusable and the destination uses the signal from the source only; the achievable rates are then limited to

$$R < \frac{1}{2}\log_2\left(1+\Gamma_{sd}\right) \triangleq I_{sd}. \tag{4.10}$$

4.2.4 Comments on the compression

We give here some comments on the constraint (4.8) that links the compression parameters N_q and R_r.

First, the range of appropriate R_r for a given N_q extends as follows :

$$\log_2\left(1+\frac{1}{N_q}\right) < R_r \leq \log_2\left(1+\frac{1+\Gamma_{sr}}{N_q}\right), \tag{4.11}$$

where the limits correspond to the cases where Γ_{sd} would be, respectively, very large or equal to 0. The relay can thus choose any R_r in this range, depending on the assumption it makes on the (unknown) Γ_{sd}.

Second, the most conservative choice of R_r, i.e. the upper limit in (4.11), reduces to using conventional rate-distortion coding at the relay instead of Wyner-Ziv coding : decompression is possible (i.e. (4.8) is satisfied) whatever the value of Γ_{sd}. In the sequel we show that this most conservative choice of R_r is often optimal in usual configurations.

4.3 OPTIMAL PARAMETERS FOR CF

A relay using CF has to choose the values of the parameters R_r and N_q. This can be done on a frame-by-frame basis after reception of $\boldsymbol{Y_{sr}}$, to take advantage of the knowledge of $\Gamma_{sr} = \gamma_{sr}$.

This section addresses the calculation of these optimal parameters so as to minimize the outage probability for known $\Gamma_{sr} = \gamma_{sr}$. This outage probability is developed in subsection 4.3.1; then the optimal value of R_r for fixed N_q is derived in subsection 4.3.2. Finding the optimal N_q however, has to be done numerically.

4.3.1 Outage probability

In the CF case, an outage event occurs when either (4.9) or (4.10) are not satisfied, depending on whether the events $\mathcal{C}$ and $\mathcal{D}$ occur. More precisely, if the decoding fails ($\bar{\mathcal{C}}$), or if it succeeds but the decompression fails ($\mathcal{C}, \bar{\mathcal{D}}$), the appropriate constraint on the spectral efficiency is (4.10); if both decoding and decompression succeed ($\mathcal{C}, \mathcal{D}$) then (4.9) is the right constraint. This yields the following decomposition of the outage probability for a spectral efficiency R and given $\Gamma_{sr} = \gamma_{sr}$:

$$\begin{aligned} & P_{\text{out}}(R|\gamma_{sr}) \qquad (4.12)\\ &= P(R \geq I_{CF}|\mathcal{C},\mathcal{D},\gamma_{sr}) \cdot P(\mathcal{D}|\mathcal{C},\gamma_{sr}) P(\mathcal{C}|\gamma_{sr}) \\ &+ P(R \geq I_{sd}) \cdot \left(P(\bar{\mathcal{C}}|\gamma_{sr}) + P(\bar{\mathcal{D}}|\mathcal{C},\gamma_{sr}) P(\mathcal{C}|\gamma_{sr})\right) \end{aligned}$$

The probabilities appearing in (4.12) are :

- The probability of not satisfying the rate constraint (4.9), given that the destination has been able to decode and decompress the signal from the relay :

$$\begin{aligned} &P(R \geq I_{CF}|\mathcal{C},\mathcal{D},\gamma_{sr}) \\ &\quad = \quad 1-\exp\left(\frac{-1}{\sigma_{sd}^2}\left(2^{2R}-1-\frac{\gamma_{sr}}{1+N_q}\right)\right). \end{aligned} \tag{4.13}$$

  This is valid only if $N_q > \frac{\gamma_{sr}}{2^{2R}-1} - 1$; this inequality always holds since we use CF only when the relay is not able to decode, thus when the right-hand member of the inequality is negative.

- The probability of not satisfying the rate constraint (4.10) :

$$P(R \geq I_{sd}) = 1-\exp\left(-\frac{2^{2R}-1}{\sigma_{sd}^2}\right). \tag{4.14}$$

- The probability of successfully decoding the signal from the relay :

$$P(\mathcal{C}|\gamma_{sr}) = \exp\left(-\frac{2^{R_r}-1}{\sigma_{rd}^2}\right). \tag{4.15}$$

- And finally the probability of successfully decompressing the signal from the relay :

$$P(\mathcal{D}|\mathcal{C},\gamma_{sr}) = \exp\left(\frac{-1}{\sigma_{sd}^2}\left(\frac{\gamma_{sr}}{(2^{R_r}-1)N_q-1}-1\right)\right) \tag{4.16}$$

  which is valid for pairs (R_r, N_q) satisfying (4.11).

4.3.2 Code rate R_r minimizing outage

This section addresses the optimal choice of the relay code rate R_r. For fixed N_q, an analytical expression of the optimal R_r is obtained by simply finding the zero of the partial derivative of (4.12) with respect to R_r.

The partial derivative $\frac{\partial P_{out}}{\partial R_r}$ can be transformed into a polynomial function of the variable 2^{R_r} after several simplifications that conserve its zeros and its sign (see appendix 4.A.1). This polynomial is

$$2^{2R_r}N_q^2 - 2^{R_r}2N_q(N_q+1) + (N_q+1)^2 - \frac{\gamma_{sr}\sigma_{rd}^2}{\sigma_{sd}^2}N_q. \tag{4.17}$$

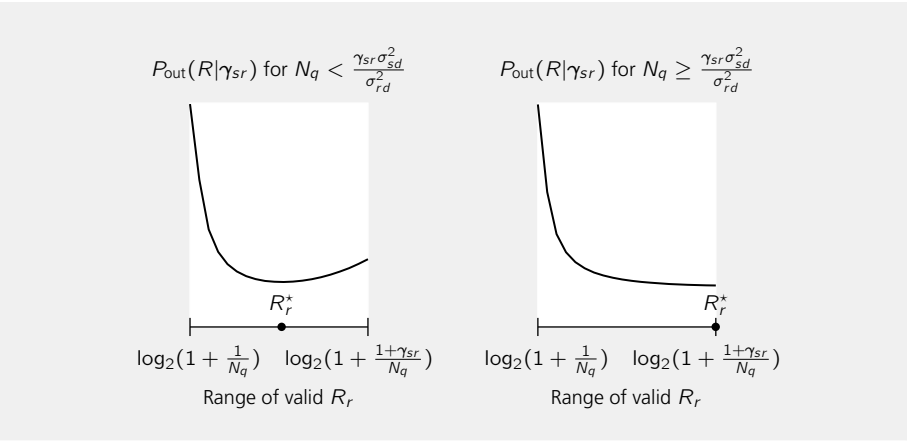


Figure 4.2: Outage probability $P_{\text{out}}(R|\gamma_{sr})$, when $N_q < \gamma_{sr}\sigma_{sd}^2/\sigma_{rd}^2$ (left) or when $N_q \geq \gamma_{sr}\sigma_{sd}^2/\sigma_{rd}^2$ (right). In the latter case, the probability of outage has a negative slope for all R_r in the range (4.11); consequently, the optimal R_r for the considered N_q is the upper limit of this range. Such a solution corresponds to usual rate-distortion coding.

Figure 4.2 illustrates what follows. For $N_q < \frac{\gamma_{sr}\sigma_{sd}^2}{\sigma_{rd}^2}$ the polynomial (4.17) has one root in the range (4.11), equal to

$$R_r^\star(N_q) = \log_2\left(1 + \frac{1}{N_q}\left(1 + \sqrt{\frac{\gamma_{sr}\sigma_{rd}^2}{\sigma_{sd}^2}N_q}\right)\right). \tag{4.18}$$

This root corresponds to the unique minimum of the probability of outage, as can be seen from (4.17).

On the contrary if $N_q \geq \frac{\gamma_{sr}\sigma_{sd}^2}{\sigma_{rd}^2}$, the polynomial (4.17) is negative for all R_r in the range (4.11), thus the outage probability decreases for increasing R_r. The optimal R_r is then the largest R_r allowed by (4.11), i.e.

$$R_r^\star(N_q) = \log_2\left(1 + \frac{1+\gamma_{sr}}{N_q}\right). \tag{4.19}$$

Among these two solutions, (4.18) corresponds to Wyner-Ziv coding, whereas (4.19) uses conventional rate-distortion coding.

4.3.3 Quantization noise variance N_q minimizing outage

The optimal solution for R_r can be substituted in the outage probability (4.12), which can then be minimized over N_q. There is however no explicit expression for the optimal

value of N_q, so that one has to resort to numerical optimization. Furthermore, the outage probability is not convex in N_q, even if we have never observed any case in which there are multiple local minima. After a thorough examination of the function, we conjecture the following :

Conjecture 1. *The outage probability* (4.12) *has only one minimum, that we write* $(R_r^\star(N_q^\star), N_q^\star)$.

4.4 BOUND ON THE USAGE OF WYNER-ZIV CODING

The aim of this section is to assess how often the optimal compression parameters $(R_r^\star(N_q^\star), N_q^\star)$ correspond to Wyner-Ziv coding, and to usual rate-distortion coding.

To do so, we first observe the derivative $\frac{dP_{out}}{dN_q}$ at the border between the regions of Wyner-Ziv coding and usual rate-distortion coding, i.e. for $N_q = \frac{\gamma_{sr}\sigma_{sd}^2}{\sigma_{rd}^2}$ (fig. 4.3). The sign of this derivative tells on which side the N_q minimizing outage is, and thus which compression method is optimal. From this observation, we identify a range of values of γ_{sr} in which the optimal solution $N_q^\star$ never corresponds to Wyner-Ziv coding. Finally this yields an upper bound on the usage of Wyner-Ziv coding.

We start with the outage probability for values of $N_q < \frac{\gamma_{sr}\sigma_{sd}^2}{\sigma_{rd}^2}$, and substitute the corresponding optimal R_r given by (4.18). The derivative $\frac{dP_{out}}{dN_q}$ can be transformed into the following function, after several simplifications that conserve its zeros and its sign (see appendix 4.A.2) :

$$\left[\exp\left(\frac{-\gamma_{sr}}{\sigma_{sd}^2(1+N_q)}\right) - 1\right]\left[\frac{1+\sqrt{\frac{\gamma_{sr}\sigma_{rd}^2}{\sigma_{sd}^2}N_q}}{\sigma_{rd}^2 N_q^2}\right] + \frac{\gamma_{sr}}{\sigma_{sd}^2(1+N_q)^2} \tag{4.20}$$

Assuming that conjecture 1 holds, the value of (4.20) when N_q gets close to the border of the Wyner-Ziv region, i.e. for $N_q \to \frac{\gamma_{sr}\sigma_{sd}^2}{\sigma_{rd}^2}$, tells in which region the optimal N_q lies :

- If (4.20) is positive, then $\frac{dP_{out}}{dN_q} > 0$ and the minimum $N_q^\star$ is such that $N_q^\star < \frac{\gamma_{sr}\sigma_{sd}^2}{\sigma_{rd}^2}$. In that case the relay uses Wyner-Ziv coding.
- If (4.20) is negative or zero, then $N_q^\star > \frac{\gamma_{sr}\sigma_{sd}^2}{\sigma_{rd}^2}$ and the relay uses conventional rate-distortion coding.

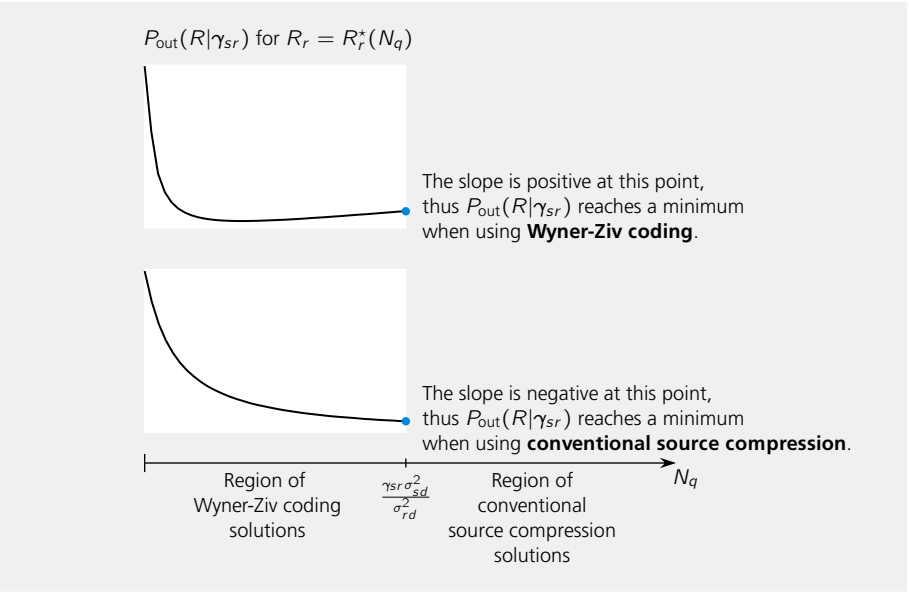


Figure 4.3: The slope of P_{out} in $N_q = \gamma_{sr}\sigma_{sd}^2/\sigma_{rd}^2$ shows which of Wyner-Ziv coding or usual rate-distortion coding is optimal.

We now find a value $\hat{\gamma}_{sr}$ such that for every $\gamma_{sr} \in [0, \hat{\gamma}_{sr}]$ the outage probability is minimized with conventional rate-distortion coding. An upper-bound of (4.20) is obtained by using $e^{-x} \leq 1 - x + \frac{x^2}{2}$. After substituting $N_q = \frac{\gamma_{sr}\sigma_{sd}^2}{\sigma_{rd}^2}$, the function can be transformed into the following second-order polynomial in γ_{sr} :

$$\gamma_{sr}^2 \left(1 + \frac{\sigma_{rd}^2}{2\sigma_{sd}^6} - \frac{1}{\sigma_{sd}^2}\right) + \gamma_{sr} \left(-\frac{\sigma_{rd}^2}{\sigma_{sd}^4} + \frac{\sigma_{rd}^2}{2\sigma_{sd}^6} - \frac{1}{\sigma_{sd}^2}\right) - \frac{\sigma_{rd}^2}{\sigma_{sd}^4}. \tag{4.21}$$

Conventional rate-distortion coding is always used when this polynomial is negative, this happens :

- If the coefficient of the first term is negative, since in that case one can check that the polynomial is negative for every γ_{sr}.
- If the coefficient of the first term is positive, the polynomial is convex and has one positive root written $\hat{\gamma}_{sr}$. The polynomial is negative for $\gamma_{sr} \in [0, \hat{\gamma}_{sr}]$ thus conventional rate-distortion coding is always used in that range of γ_{sr}.

Now we can calculate an upper-bound on the usage of Wyner-Ziv coding. As discussed above, Wyner-Ziv is never used if $\gamma_{sr} < \hat{\gamma}_{sr}$, and CF in general is never used if $\gamma_{sr} >$

$2^{2R}-1$. Thus, at most, CF with Wyner-Ziv coding can be used for $\gamma_{sr} \in [\hat{\gamma}_{sr}, 2^{2R}-1[$. The statistics of the source-to-relay channel then determine how often this happens. This yields the following upper bound on the probability of using Wyner-Ziv compress-and-forward :

$$P(\hat{\gamma}_{sr} < \gamma_{sr} \leq 2^{2R}-1) = \exp\left(-\frac{\hat{\gamma}_{sr}}{\sigma_{sr}^2}\right) - \exp\left(-\frac{2^{2R}-1}{\sigma_{sr}^2}\right). \tag{4.22}$$

Note that increasing the rate R does not change the lower limit of the Wyner-Ziv region $\hat{\gamma}_{sr}$ but increases the upper limit; thus Wyner-Ziv coding is more often used for higher rates R.

4.5 SIMULATIONS

This section illustrates the results developed above with an investigation of the usage of Wyner-Ziv coding for several choices of the transmission powers and spectral efficiencies. We show that for low or moderate spectral efficiencies R, being limited to conventional rate-distortion coding does not have a significant impact on the outage probability.

The relay network is depicted in fig. 4.1 : all three terminals are lined up, the relay is at a distance d from the source and the distance between the source and the destination is normalized to 1. Path loss between the terminals determines the power levels at the receivers. With a path loss exponent chosen equal to $\alpha = 3.5$ we have $\sigma_{sd}^2 = P_s$, $\sigma_{sr}^2 = P_s/d^\alpha$ and $\sigma_{rd}^2 = P_r/(1-d)^\alpha$.

The relay uses DF if it is able to decode, otherwise it uses CF. The parameters of CF are optimized, and correspond either to Wyner-Ziv coding or to conventional rate-distortion coding.

Based on the results of section 4.4, we evaluate the percentage of situations in which DF and CF are used, for each position d of the relay; then among situations requiring CF, we evaluate the upper bound (4.22) on the proportion of them requiring Wyner-Ziv coding.

Figures 4.4–4.6 present three situations with different spectral efficiencies $R = 0.5$, 1 and 2 b/s/Hz respectively, with a choice of transmission powers P_s and P_r such that the overall outage probability is below or around 10^{-2} in each case. The upper subgraphs show the usage of each CF method as a function of the position of the relay d, both theoretically (filled areas) or as observed in Monte-Carlo simulations (crosses

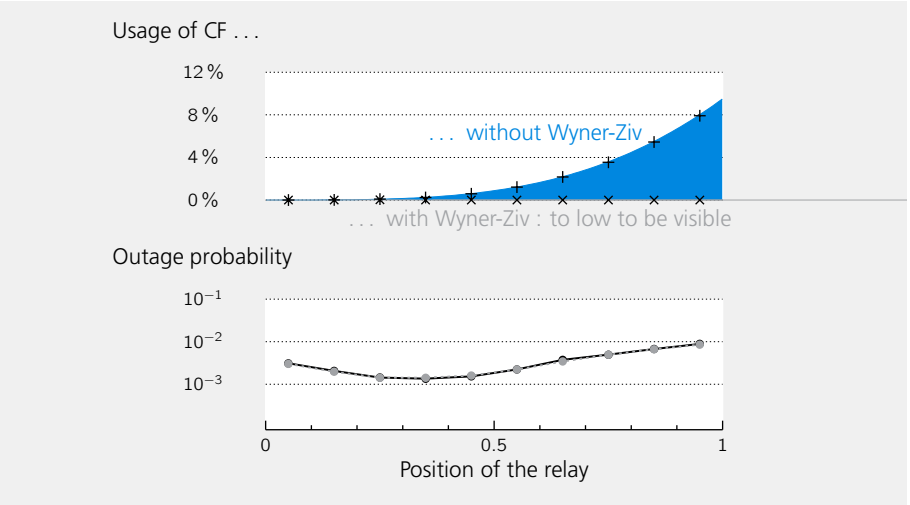


Figure 4.4: Usage of the CF protocols for $R = 0.5$, $P_s = P_r = 10$. The optimal compression does almost never require Wyner-Ziv (usage lower than 0.02 %). In the upper sub-graphs the colored areas correspond to theoretical results, crosses + and x correspond to simulations. The outage curves in the lower sub-graphs are obtained by simulation.

+ and $\times$). Lower sub-graphs show the corresponding outage probability as obtained by simulation. The dashed curves are obtained when limiting the optimization of relay parameters to conventional rate-distortion coding.

In figure 4.4, for $R = 0.5$ b/s/Hz and $P_s = P_s = 10$ J/symbol, the optimal compression method is almost never Wyner-Ziv coding (usage in less than 0.02 % of the frames at its maximum). Note that this case corresponds to $R_s = 1$ b/s/Hz. Figure 4.5 shows that for $R = 1$ b/s/Hz and $P_s = P_s = 25$ J/symbol, Wyner-Ziv is used in up to 1 % of the cases, for some positions of the relay. However, the outage curves show almost no difference if conventional rate-distortion coding is always used. Finally, figure 4.6 shows that for $R = 2$ b/s/Hz and $P_s = P_s = 100$ J/symbol, Wyner-Ziv coding is more often used and brings a performance gain.

These results highlight first that Wyner-Ziv is beneficial only if the source-to-relay and source-to-destination signals are received with a good signal-to-noise ratio, so that their correlation can be exploited. The bound of section 4.4 already showed this : Wyner-Ziv coding is used only if γ_{sr} is larger than a threshold $\hat{\gamma}_{sr}$, whereas conventional rate-distortion coding is used for smaller values of γ_{sr}.

Second, these results show that the use of Wyner-Ziv coding decreases when the relay gets close to the destination; this is because the relay-to-destination channel is

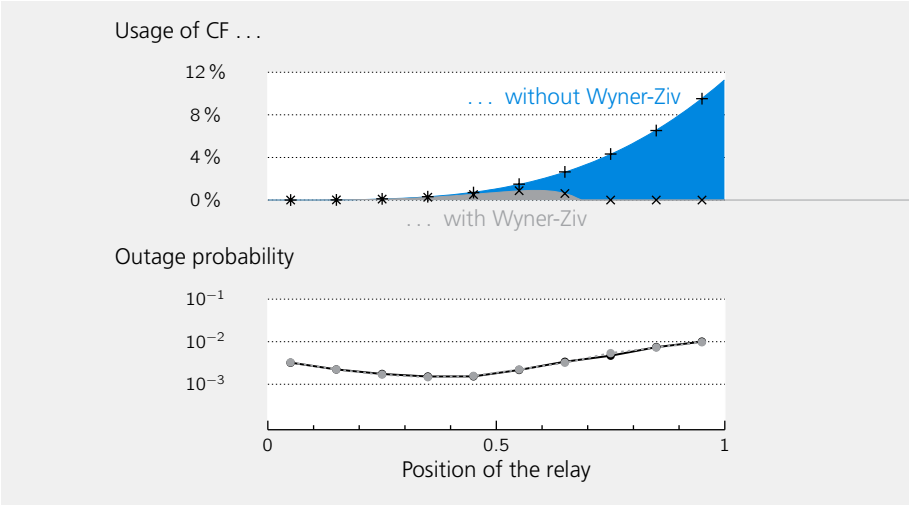


Figure 4.5: Usage of the CF protocols for $R = 1$, $P_s = P_r = 25$. Wyner-Ziv coding is optimal for up to 1 % of the frames, but using conventional rate-distortion coding instead does barely have any impact on the outage (dotted curve in lower graph). In the upper sub-graphs the colored areas correspond to theoretical results, crosses + and x correspond to simulations. The outage curves in the lower sub-graphs are obtained by simulation.

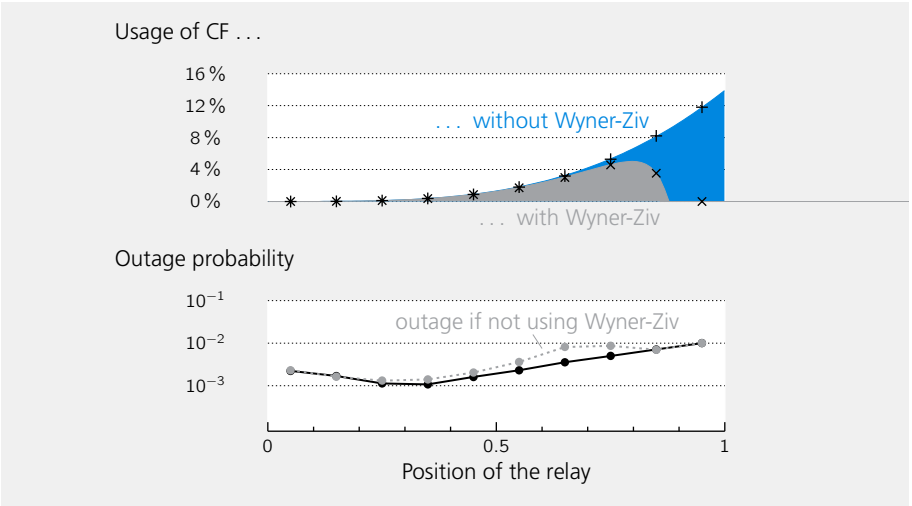


Figure 4.6: Usage of the CF protocols for $R = 2$, $P_s = P_r = 100$. Wyner-Ziv coding is more often used than in fig. 4.4 and 4.5; not using it has an impact on the outage. In the upper sub-graphs the colored areas correspond to theoretical results, crosses + and x correspond to simulations. The outage curves in the lower sub-graphs are obtained by simulation.

expected to be strong, thus R_r can be chosen large. The availability of a high trans-

mission rate R_r makes Wyner-Ziv coding less appealing.

4.6 DISCUSSION

In the CF protocol, the destination does not jointly decode the signals from the source and the relay. This makes the overall performance dependent on the quality of point-to-point channels. First, the point-to-point relay-to-destination transmission must be successful (constraint (4.7)). Then, the point-to-point source-to-destination channel must be good enough to enable successful Wyner-Ziv decoding (constraint (4.8)). Only after these two preliminary steps, the destination can decode the source message by recombining the source and relay signals (rate (4.9)).

As opposed to CF, the AF protocol make the destination decode the source and relay signals jointly. This gives an advantage to the AF protocol over fading channels without transmit CSI, even though the AF rate is lower than the CF rate on static channels.

For this reason, we compare our adaptive DF-CF protocol to the hybrid DF-AF protocol in the following two sections.

4.6.1 Comparison with an adaptive DF-AF protocol

Uncoded (analog) transmission of a Gaussian source on a Gaussian channel is known to achieve optimal mean square error distortion [58]. In comparison with digital coded transmission of the same source, such an analog transmission has the advantage not to need transmit CSI to achieve optimal performance.

Similarly, since in our case the signal to compress at the relay is Gaussian, the AF protocol yields the same distortion at the destination as the CF protocol with conventional rate-distortion coding when the relay has transmit CSI. However when the relay has no transmit CSI, CF cannot perform as well as AF.

This conclusion is valid only for CF with conventional rate-distortion coding, and not for CF with Wyner-Ziv coding. But as shown above, rate-distortion coding might be much more often used than Wyner-Ziv coding in an adaptive CF system. In the following we compare by simulation the outage of the DF-CF scheme of this chapter with an hybrid DF-AF scheme.

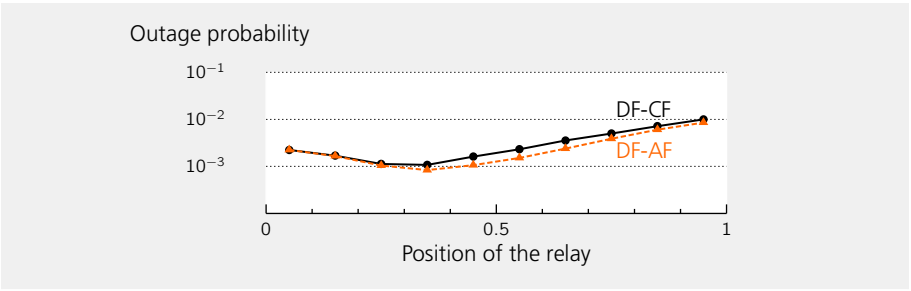


Figure 4.7: Without transmit CSI the hybrid DF-AF protocol outperforms the DF-CF protocol, even though CF achieves a higher rate than AF on static channels. $P_s = P_r = 100$, $R = 2$.

In the AF protocol, the relay transmits the signal

$$\boldsymbol{X_r} = \boldsymbol{Y_{sr}} \sqrt{\frac{P_r}{|h_{sr}|^2 P_s + N_{sr}}}. \tag{4.23}$$

Achievable spectral efficiencies R satisfy the constraint

$$R < \frac{1}{2} \log_2 \left(1 + \Gamma_{sd} + \frac{\gamma_{sr} \Gamma_{rd}}{1 + \gamma_{sr} + \Gamma_{rd}} \right). \tag{4.24}$$

The simulation results in fig. 4.7 show that the DF-AF scheme achieves better performance. As suggested above, with no transmit CSI the relay often chooses conventional rate-distortion coding instead of Wyner-Ziv coding, and CF with conventional rate-distortion coding is worse than AF. While these simulation results cannot claim to be general, we include them here because we have never observed a case in which DF-CF achieves a lower outage than DF-AF.

Note that these conclusions are particular to the choices made in this chapter and do not necessarily extend, for instance, to situations where the time/bandwith are not equally shared between the source and the relay, or to practical (not Gaussian) transmission schemes.

4.6.2 Comparison with an adaptive DF-AF protocol, when the relay knows the level of side information

In this chapter we have shown that when the relay has only receive CSI, it is often better to use conventional rate-distortion coding instead of Wyner-Ziv compression. This is because the success of Wyner-Ziv decoding depends on Γ_{sd} (equation (4.8)), which is

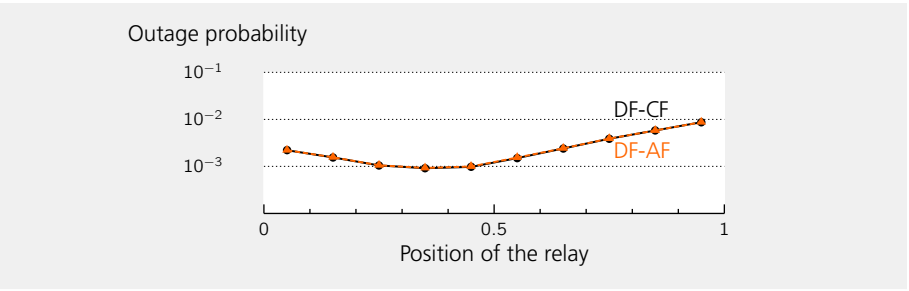


Figure 4.8: When the relay knows the level of side information available at the destination, the DF-CF protocol outperforms the DF-AF protocol but the difference is so small that it is not visible on this figure. $P_s = P_r = 100$, $R = 2$.

unknown to the relay.

However, the risk of incorrect Wyner-Ziv decoding can be eliminated if the destination informs the relay of Γ_{sd}. This can be done at the end of the first slot, once the destination has observed $\boldsymbol{Y_{sd}}$; consequently the relay can choose N_q and R_r that satisfy both the rate constraint (4.9) and the decompression constraint (4.8).

As shown in appendix 4.B, the achievable rate for CF when the relay knows $\Gamma_{sd} = \gamma_{sd}$ is

$$R < \frac{1}{2}\log_2\left(1 + \gamma_{sd} + \frac{\gamma_{sr}\Gamma_{rd}}{1 + \frac{\gamma_{sr}}{1+\gamma_{sd}} + \Gamma_{rd}}\right). \tag{4.25}$$

This rate is always higher than the AF rate (4.24), even though only slightly so. Note that the same rate was found in [59, chapter 5] for a CF relay with full CSI (i.e., when the relay knows Γ_{sr}, Γ_{sd} and also Γ_{rd}).

Fig. 4.8 compares the outage probabilities with each of these protocols; the difference is too small to be visible on the figure.

Again, this conclusion does not necessarily extend to practical transmission schemes, for which compression minimizing mean square error distortion is not optimal.

4.7 CONCLUSION

This chapter addresses the optimization of the Wyner-Ziv coding parameters of the CF protocol, and shows that for a range of transmission rates and powers the compression method minimizing outage is simply conventional rate-distortion coding. Being able to resort to this simpler compression method without significant performance gain

is particularly interesting for the design and implementation of practical transmission systems with CF relays.

We have also compared the performance of the hybrid DF-CF protocol with an hybrid DF-AF protocol; if the relay knows the level of side information available at the destination DF-CF can achieve a slightly lower outage probability than DF-AF, otherwise we have observed that DF-AF performs better.

These results are valid for the Gaussian relay channel with Gaussian codes; they do not necessarily extend to practical CF codes even though they are a priori relevant in that case as well. Also, if a joint decoding method was found for the CF protocol, the preceding results regarding the usage of Wyner-Ziv compression would no longer hold and the performance of CF over fading channels without transmitter CSI would improve.

Appendix 4.A DERIVATIVES OF THE OUTAGE PROBABILITY

4.A.1 Partial derivative with respect to R_r

In this section we calculate the partial derivative of $P_{\text{out}}(R|\gamma_{sr})$, given in (4.12), with respect to the rate R_r.

We first note that $P(\mathcal{C}|\gamma_{sr}) = 1 - P(\bar{\mathcal{C}}|\gamma_{sr})$ involves

$$\frac{\partial}{\partial R_r} P(\mathcal{C}|\gamma_{sr}) = -\frac{\partial}{\partial R_r} P(\bar{\mathcal{C}}|\gamma_{sr}). \tag{4.26}$$

and similarly for $P(\mathcal{D}|\mathcal{C}, \gamma_{sr})$.

Using these, the partial derivative of $P_{\text{out}}(R|\gamma_{sr})$ can be written

$$\begin{aligned} &\frac{\partial}{\partial R_r} P_{\text{out}}(R|\gamma_{sr}) \\ &= -\left(P(R \geq I_{sd}) - P(R \geq I_{CF}|\mathcal{C}, \mathcal{D}, \gamma_{sr})\right) \\ &\times \left(P(\mathcal{D}|\mathcal{C}, \gamma_{sr}) \frac{\partial}{\partial R_r} P(\mathcal{C}|\gamma_{sr}) + P(\mathcal{C}|\gamma_{sr}) \frac{\partial}{\partial R_r} P(\mathcal{D}|\mathcal{C}, \gamma_{sr})\right). \end{aligned} \tag{4.27}$$

The derivatives appearing above are equal to

$$\frac{\partial}{\partial R_r} P(\mathcal{C}|\gamma_{sr}) = P(\mathcal{C}|\gamma_{sr}) \frac{-2^{R_r} \log(2)}{\sigma_{rd}^2} \tag{4.28}$$

and

$$\frac{\partial}{\partial R_r} P(\mathcal{D}|\mathcal{C}, \gamma_{sr}) = P(\mathcal{D}|\mathcal{C}, \gamma_{sr}) \frac{2^{R_r} \log(2) \gamma_{sr} N_q}{\sigma_{sd}^2((2^{R_r}-1)N_q - 1)^2}. \tag{4.29}$$

As we are interested only in finding the roots and the sign of the derivative (4.27), we now remove all factors that are always positive. The first factor in (4.27) is always positive (behalves if $\gamma_{sr} = 0$, but this event has probability 0), and can thus be neglected. The second factor can be divided by $P(\mathcal{C}|\gamma_{sr})P(\mathcal{D}|\mathcal{C}, \gamma_{sr})$ and by $2^{R_r}\log(2)$. This gives the following function, that has the same roots and the same sign as the derivative (4.27) :

$$\frac{1}{\sigma_{rd}^2} - \frac{\gamma_{sr} N_q}{\sigma_{sd}^2((2^{R_r}-1)N_q - 1)^2}. \tag{4.30}$$

Finally, multiplying by the positive factor $\sigma_{rd}^2((2^{R_r}-1)N_q - 1)^2$ yields the polynomial (4.17).

4.A.2 Derivative with respect to N_q

This section presents the calculation of the derivative of $P_{\text{out}}(R|\gamma_{sr})$ with respect to N_q, for $N_q < \frac{\gamma_{sr}\sigma_{sd}^2}{\sigma_{rd}^2}$ and the optimal $R_r = R_r^\star(N_q)$ given by (4.18).

After substituting $R_r = R_r^\star(N_q)$, the probabilities (4.15) and (4.16) become

$$P(\mathcal{C}|\gamma_{sr}) = \exp\left(\frac{-1}{\sigma_{rd}^2 N_q}\left(1 + \sqrt{\frac{\gamma_{sr}\sigma_{rd}^2}{\sigma_{sd}^2} N_q}\right)\right) \tag{4.31}$$

$$P(\mathcal{D}|\mathcal{C}, \gamma_{sr}) = \exp\left(\frac{-1}{\sigma_{sd}^2}\left(\sqrt{\frac{\gamma_{sr}\sigma_{sd}^2}{\sigma_{rd}^2 N_q}} - 1\right)\right). \tag{4.32}$$

Their derivatives with respect to N_q are

$$\frac{d}{dN_q} P(\mathcal{C}|\gamma_{sr}) = \frac{P(\mathcal{C}|\gamma_{sr})}{\sigma_{rd}^2 N_q^2}\left(\sqrt{\frac{\gamma_{sr}\sigma_{rd}^2}{4\sigma_{sd}^2} N_q} + 1\right) \tag{4.33}$$

$$\frac{d}{dN_q} P(\mathcal{D}|\mathcal{C}, \gamma_{sr}) = P(\mathcal{D}|\mathcal{C}, \gamma_{sr}) \sqrt{\frac{\gamma_{sr}}{4\sigma_{sd}^2\sigma_{rd}^2 N_q^3}}. \tag{4.34}$$

In the following we also need the derivative

$$\frac{d}{dN_q}P(R \geq I_{CF}|\mathcal{C},\mathcal{D},\gamma_{sr}) = \\ (P(R \geq I_{CF}|\mathcal{C},\mathcal{D},\gamma_{sr}) - 1)\left(\frac{-\gamma_{sr}}{\sigma_{sd}^2(1+N_q)^2}\right). \tag{4.35}$$

These preliminary steps enable now the calculation of the derivative of $P_{\text{out}}(R|\gamma_{sr})$:

$$\begin{aligned} &\frac{d}{dN_q}P_{\text{out}}(R|\gamma_{sr}) \\ &= (P(R \geq I_{CF}|\mathcal{C},\mathcal{D},\gamma_{sr}) - P(R \geq I_{sd})) \\ &\times \left(P(\mathcal{D}|\mathcal{C},\gamma_{sr})\frac{d}{dN_q}P(\mathcal{C}|\gamma_{sr}) + P(\mathcal{C}|\gamma_{sr})\frac{d}{dN_q}P(\mathcal{D}|\mathcal{C},\gamma_{sr})\right) \\ &+ \ P(\mathcal{C}|\gamma_{sr})P(\mathcal{D}|\mathcal{C},\gamma_{sr})\frac{d}{dN_q}P(R \geq I_{CF}|\mathcal{C},\mathcal{D},\gamma_{sr}). \end{aligned} \tag{4.36}$$

We substitute the derivatives (4.33), (4.34) and (4.35) into (4.36), and again, as we are interested only in the roots and the sign of the derivative, we remove factors that are always positive. We divide thus (4.36) by $P(\mathcal{C}|\gamma_{sr})$, $P(\mathcal{D}|\mathcal{C},\gamma_{sr})$ and $(1 - P(R > I_{CF}|\mathcal{C},\mathcal{D},\gamma_{sr}))$ and finally obtain the function (4.20).

Appendix 4.B CF RATE WITH KNOWN LEVEL OF SIDE INFORMATION

If the relay knows both $\Gamma_{sr} = \gamma_{sr}$ and $\Gamma_{sd} = \gamma_{sd}$, it can choose compression parameters N_q and R_r such that the constraints (4.8) and (4.9) are always satisfied. The success of the CF transmission becomes dependent on the inequality (4.7) only, i.e., on the success of the transmission on the relay-to-destination channel.

More specifically, the parameters N_q and R_r are chosen as follows. The compression noise N_q is chosen so as to satisfy the rate constraint (4.9), thus

$$N_q^\star = \frac{\gamma_{sr}}{2^{2R} - 1 - \gamma_{sd}} - 1. \tag{4.37}$$

This value is always positive, behalves in situations we discard because relaying is not needed :

- The value is negative or infinite if $2^{2R} - 1 - \gamma_{sd} \leq 0$, or equivalently if $R \leq \frac{1}{2}\log_2(1+\gamma_{sd})$, but in in this case the destination is able to decode without help

from the relay.

- The value is negative or zero if $\gamma_{sr} \leq 2^{2R} - 1 - \gamma_{sd}$, or equivalently if $R \geq \frac{1}{2}\log_2(1 + \gamma_{sd} + \gamma_{sr})$, but in this case even perfect transmission from the relay to the destination would not enable successful decoding.

Next the rate R_r is chosen as the smallest rate satifying (4.8) :

$$R_r^\star = \log_2\left(1 + \frac{1 + \frac{\gamma_{sr}}{1+\gamma_{sd}}}{N_q^\star}\right). \tag{4.38}$$

Given these choices $N_q^\star$ and $R_r^\star$, the inequality (4.7) alone determines the success of the overall relayed transmission :

$$R_r^\star < \log_2(1 + \Gamma_{rd}). \tag{4.39}$$

This constraint can be expressed equivalently as a constraint on the overall transmission rate R, as we show now. Substituting (4.38) into (4.39) gives the inequality

$$\Gamma_{rd} > \frac{1 + \frac{\gamma_{sr}}{1+\gamma_{sd}}}{N_q^\star}. \tag{4.40}$$

Multiplying by $N_q^\star$ and substituting its value given by (4.37) yields

$$\frac{\gamma_{sr}\Gamma_{rd}}{2^{2R} - 1 - \gamma_{sd}} - \Gamma_{rd} > 1 + \frac{\gamma_{sr}}{1 + \gamma_{sd}} \tag{4.41}$$

Next we multiply by $2^{2R} - 1 - \gamma_{sd}$ (always positive in the case we consider) and re-arrange the terms to find

$$\gamma_{sr}\Gamma_{rd} > (2^{2R} - 1 - \gamma_{sd})\left(1 + \frac{\gamma_{sr}}{1 + \gamma_{sd}} + \Gamma_{rd}\right). \tag{4.42}$$

Finally, making R explicit gives

$$R < \frac{1}{2}\log_2\left(1 + \gamma_{sd} + \frac{\gamma_{sr}\Gamma_{rd}}{1 + \frac{\gamma_{sr}}{1+\gamma_{sd}} + \Gamma_{rd}}\right). \tag{4.43}$$

This rate is slightly higher than the AF rate given in (4.24); it is achieved with CF when the relay knows the level of side information available at the destination, γ_{sd}.

Practical codes and decoders for the relay channel

5

The previous two chapters have addressed the relay channel from an information-theoretic point of view. In this chapter, we focus instead on practical (in the sense *implementable*) coded systems for transmission on the relay channel.

The chapter starts by describing implementations of the DF and AF protocols, then presents three original contributions of this thesis. These contributions share a common objective : exploiting signals that are not reliable enough to be decoded correctly at the relay. The first contribution is termed Soft-DF; it aims at regenerating (i.e., decoding) the signal at the relay and forwarding it in a way that conveys information over its reliability. The second one is a modified DF receiver, that can deal with errors made by the relay. The third contribution is an iterative MAP receiver for practical implementations of the CF protocol.

Finally, all protocols are compared when used in hybrid schemes over wireless relay channels. From these comparisons, the most promising technique is probably the CF protocol with the associated MAP receiver.

The contributions of this chapter were first reported in [5, 6, 7, 8, 9].

Contents

5.1 INTRODUCTION

This chapter addresses practical (implementable) coded systems for relaying.

Several code designs for the DF protocol have been proposed in the literature. For relaying on orthogonal channels, the contributions [60, 61] propose designs based on rate-compatible punctured convolutional codes [62]. In these schemes, the source transmits with a punctured convolutional code and the relay, after decoding, forwards additional parity bits. By adding an interleaver at the relay, one obtains a distributed turbo-code (see [63, 64]) : even though the source terminal transmits a convolutional code, the signals from the source terminal and from the relay make up a turbo-code. More recently, the contribution [65] has proposed a code design based on LDPC codes. Code designs have also been proposed for the cases of half-duplex or full-duplex relay channels. They are based either on turbo-codes (see [66, 67]) or on LDPC codes (see [68,69,70,71,72]). Other contributions include code designs with both channel coding and network coding, as in [73].

The DF protocol requires the relay to decode successfully the signal from the source terminal. When the relay cannot decode the signal, non-regenerative protocols like AF or CF must be considered. Coded implementations of these protocols have been less investigated, although several code designs for the CF protocol have been developed recently. They all use scalar quantization at the relay (as in [74]), possibly followed by source coding (as in [69, 75]) or Slepian-Wolf source coding (see [76]).

This chapter focuses on the cases where the relay is not able to decode the signal from the source terminal. It proposes encoding techniques and reception techniques to deal with imperfect decoding at the relay.

The propositions of this chapter are based on the distributed turbo-coded DF protocol of Valenti et al. [63], chosen for its modularity and its low-complexity at the relay. They extend Valenti's DF scheme to handle situations where decoding fails at the relay.

A first proposition, that we term Soft Decode-Forward (Soft-DF), deals with uncertainty at the relay by performing all relay operations in a soft fashion, so as to keep the reliability information [5, 6, 77, 8]. The soft values are forwarded analogically in a way that conveys their reliability, instead of using discrete symbols. The aim is to exploit the error-correcting code as much as possible (to regenerate the signal), without losing information about the uncertainty of the data.

Another possibility is to deal with relay errors on the receiver side. To do so, we propose

a decoding algorithm resilient to relay errors, i.e. a modified DF decoding algorithm that takes into account the probability of errors between the source and the relay [7]. Results show that this receiver manages to take advantage of relayed frames even if they contain errors.

After these two DF-based contributions, the third proposition of this chapter uses the CF protocol. The focus is on the receiver at the destination, for a CF system similar to the one in [76]. The receivers found in the literature for practical implementations of the CF protocol follow the lines of the information-theoretic CF decoder : they first decode the signal from the relay, decompress it and use it to help decoding the signal from the source terminal. We derive instead a MAP receiver performing joint decoding of the source and relay signals. Important differences between this receiver and the previous ones include the fact that it is iterative, and that the error-correcting code helps refining the side information used for the decompression of the relay signal. We show that this receiver achieves better performance than usual algorithms.

All these non-regenerative protocols are compared on wireless channels in the context of hybrid transmission schemes, that use DF when the relay is able to decode and a non-regenerative protocol otherwise (AF, CF, Soft-DF or DF with the modified error-resilient receiver). Interestingly, all protocols proposed in this chapter outperform the AF protocol. This contrasts with the information-theoretic case addressed in section 4.6.1, where the AF protocol was shown to perform better than the CF protocol under similar conditions. The CF protocol almost always yields the best performance on frames that the relay is not able to decode correctly.

Section 5.2 defines the model of the system under consideration, then sections 5.3 and 5.4 present, respectively, the distributed turbo-coded DF protocol and an AF protocol used as reference for comparisons. The Soft-DF protocol is presented in section 5.5, and the DF decoding algorithm resilient to relay errors in section 5.6. Then section 5.7 presents the derivation of the iterative MAP receiver for the CF protocol, and finally section 5.8 compares all protocols when used in hybrid schemes.

5.2 GENERAL SYSTEM MODEL

This chapter considers an orthogonal relay model, as presented in chapter 3. Transmissions of the source terminal and of the relay occur on channels separated in time or frequency. We refer to these separate channel uses as *slots* in the following : the source transmits in the first slot, and the destination in the second one.

Differing from chapter 3, the codes used here are not Gaussian codes. The coding operation is instead separated into binary coding followed by modulation into real symbols. Note that the case of complex (two-dimensional) symbols is a straightforward extension, not considered here.

Baseband-equivalent discrete-time model Transmission channels are modeled as frequency-flat additive white Gaussian noise channels. Using a baseband-equivalent discrete-time model of the transmissions, the signals received by the relay and the destination are written :

$$\boldsymbol{Y_{sr}} = H_{sr}\,\boldsymbol{S_s} + \boldsymbol{Z_{sr}}, \tag{5.1}$$

$$\boldsymbol{Y_{sd}} = H_{sd}\,\boldsymbol{S_s} + \boldsymbol{Z_{sd}}, \tag{5.2}$$

$$\boldsymbol{Y_{rd}} = H_{rd}\,\boldsymbol{S_r} + \boldsymbol{Z_{rd}}, \tag{5.3}$$

which describe the transmission of symbols $\boldsymbol{S_s}$ and $\boldsymbol{S_r}$ with variance $P_s \triangleq \mathrm{E}\left\{|S_s|^2\right\}$ and $P_r \triangleq \mathrm{E}\left\{|S_r|^2\right\}$ respectively, experiencing channel factors H_{sr}, H_{sd} or H_{rd} and additive noise at the reception $\boldsymbol{Z_{sr}}$, $\boldsymbol{Z_{sd}}$ or $\boldsymbol{Z_{rd}}$. These noise terms are modeled as zero-mean white Gaussian noise samples with two-sided power spectral densities $N_0/2$.

As in previous chapters, when wireless channels are considered, the fading coefficients H_{sr}, H_{sd} and H_{rd} are modeled as independent Rayleigh-distributed samples.

Each of the following sections particularizes this general system model to several protocols, codes and modulations.

5.3 DECODE-FORWARD WITH DISTRIBUTED TURBO-CODES

We consider the orthogonal relay channel model, with a source terminal transmitting to a destination with help from a relay (see fig. 5.1).

The DF protocol with distributed turbo-codes [63] consists in using convolutional codes at the source terminal and at the relay, in such a way that they make up a turbo-code at the destination. The aim is to benefit from the performance of turbo-codes while keeping the complexity low at the relay : the relay needs a BCJR decoder and not the more complex turbo-decoder.

The source terminal transmits a sequence of information bits $\boldsymbol{U_s}$ (length K), encoded by an RSC code into a sequence of coded bits $\boldsymbol{X_s}$. These bits are then modulated into

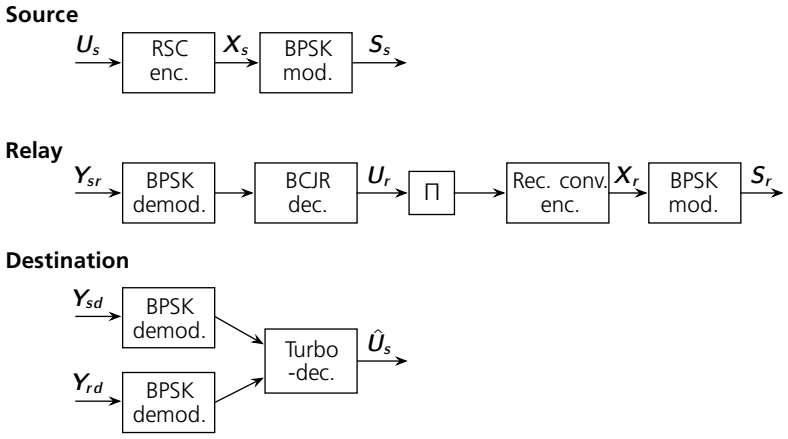


Figure 5.1: System model for the DF protocol with distributed turbo-codes.

a sequence of symbols $\boldsymbol{S_s}$ (length N) and sent over the channel. The relay receives $\boldsymbol{Y_{sr}}$ and the destination $\boldsymbol{Y_{sd}}$.

The relay demodulates and decodes $\boldsymbol{Y_{sr}}$ into bits $\boldsymbol{U_r}$. To avoid error propagation, the relay forwards the sequence $\boldsymbol{U_r}$ only if it is error-free, i.e., if $\boldsymbol{U_r} = \boldsymbol{U_s}$. This is easily checked with an error-detecting code, for example a cyclic redundancy check code [78]. This modification to the DF protocol is usually referred to as *selective* DF; we use it by default.

To transmit the bits $\boldsymbol{U_r}$, the relay interleaves them and encodes them with a recursive *non-systematic* convolutional code. The resulting parity bits $\boldsymbol{X_r}$ are modulated into symbols $\boldsymbol{S_r}$ and sent to the destination. The destination receives $\boldsymbol{Y_{rd}}$.

Finally, the destination uses both the direct and relayed signals $\boldsymbol{Y_{sd}}$ and $\boldsymbol{Y_{rd}}$ to make a joint decision. The signal from the source terminal contains both systematic bits and parity bits, and the signal from the relay contains an additional set of parity bits. Thanks to the interleaver at the relay, these bits all together make up a turbo-code. The appropriate decoding algorithm is thus the turbo-decoding algorithm (section 2.3.2).

5.4 AMPLIFY-FORWARD REFERENCE SYSTEM

We define the AF system of fig. 5.2, for comparison with other relay systems of this chapter.

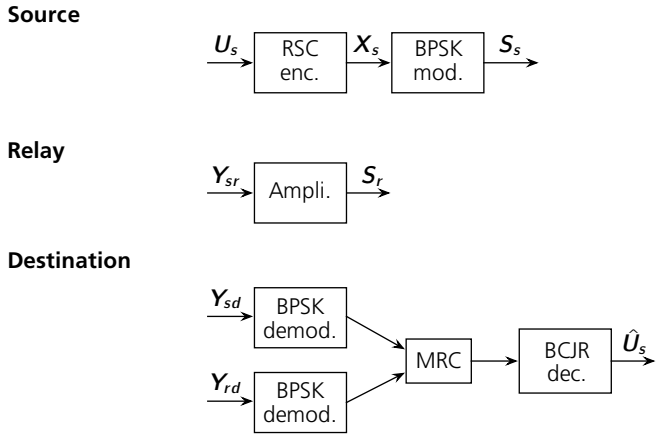


Figure 5.2: System model for the AF protocol.

This system also uses an RSC code at the source terminal. The behavior of the relay changes however. Instead of trying to decode, the relay simply amplifies the signal $\boldsymbol{Y_{sr}}$ it receives from the source. The amplification factor is such that the variance of transmitted symbols is P_r, as in the DF system of the previous section. The signal sent by the relay is :

$$\boldsymbol{S_r} = \sqrt{\frac{P_r}{|H_{sr}|^2 P_s + N_0/2}}\,\boldsymbol{Y_{sr}} \tag{5.4}$$

$$= \sqrt{\frac{P_r}{|h_{sr}|^2 P_s + N_0/2}}(H_{sr}\boldsymbol{S_s} + \boldsymbol{Z_{sr}}). \tag{5.5}$$

The destination first combines $\boldsymbol{Y_{sd}}$ and $\boldsymbol{Y_{rd}}$ with a maximum-ratio combiner; then it demodulates the combined sequence, and decodes the RSC code to obtain $\boldsymbol{U_s}$.

5.5 SOFT DECODE-FORWARD

This section presents a technique that aims at combining the main advantages of both DF and AF : it regenerates the signal as DF, but keeps soft information as AF. This technique, that we call Soft-DF, amounts to a DF scheme where all operations are performed in a soft-input soft-output fashion (fig. 5.3). Using soft values instead of taking hard decisions at the relay enables to convey reliability information in the forwarded signal.

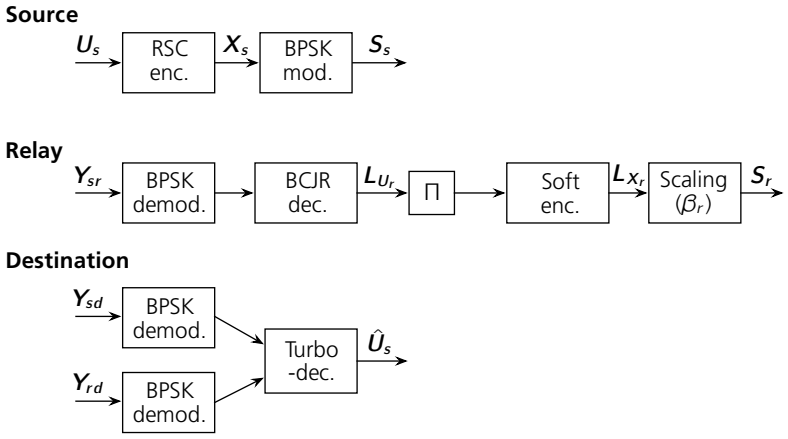


Figure 5.3: System model for the Soft-DF protocol. The variables $\boldsymbol{L}_{U_r}$ and $\boldsymbol{L}_{X_r}$ are the sequences of soft log-ratios produced by the soft decoder and soft encoder, respectively.

A soft-DF relay first decodes its partner's signal, keeping soft information, then interleaves and re-encodes it with a soft-input soft-output (SISO) encoder. The outputs of the SISO encoder can be seen as the outputs of an equivalent source-to-relay channel, with a better signal-to-noise ratio (SNR) than the actual channel. The destination is made of a turbo-decoder, as in the DF protocol of section 5.3.

We successively describe below the new SISO encoder, the equivalent channel modeling its output and finally the receiver at destination.

5.5.1 Soft-Input Soft-Output Encoder

The proposed SISO *encoding* algorithm has some similarities with a SISO *decoding* algorithm.

Given an observed signal $\boldsymbol{Y_{sr}} = \boldsymbol{y_{sr}}$, a DF relay would compute estimates $\boldsymbol{U_r} = [U_{r,1}, \ldots, U_{r,K}]^T$ of the K information bits and from these produce a sequence of N coded bits $\boldsymbol{X_r} = [X_{r,1}, \ldots, X_{r,N}]^T$. In contrast, the soft-DF relay keeps the soft information on the information bits, i.e. the a posteriori probabilities $P\big(U_{r,k}|\boldsymbol{y_{sr}}\big)$ and produces from these a sequence of a posteriori probabilities $P\big(X_{r,n}|\boldsymbol{y_{sr}}\big)$ on the coded bits $X_{r,n}$. For the sake of notational simplicity in this section, we assume that the encoder produces only one parity bit per information bit (i.e. $N = K$). Generalization to other rates is straightforward.

For a convolutional code, the representation of this soft encoding operation within the framework of factor graphs and the application of the sum-product algorithm lead to an *encoding* algorithm quite similar to the well-known BCJR *decoding* algorithm. More precisely, the soft encoder aims at computing

$$P(X_{r,k}|\boldsymbol{y_{sr}}) = \sum_{\sim X_{r,k}} P(\boldsymbol{X_r}|\boldsymbol{y_{sr}}) \qquad k = 1, \ldots, K, \tag{5.6}$$

where $\sum_{\sim X_{r,k}}$ denotes the summation on all variables but $X_{r,k}$. A convenient factorization of $P(\boldsymbol{X_r}|\boldsymbol{y_{sr}})$ must be found to enable an efficient computation of the marginals $P(X_{r,k}|\boldsymbol{y_{sr}})$ with the sum-product algorithm. Since $\boldsymbol{Y_{sr}} - \boldsymbol{U_r} - \boldsymbol{X_r}$ forms a Markov chain we have :

$$P(X_{r,k}|\boldsymbol{y_{sr}}) = \sum_{\sim X_{r,k}} P(\boldsymbol{X_r}|\boldsymbol{U_r}) \cdot P(\boldsymbol{U_r}|\boldsymbol{y_{sr}}), \tag{5.7}$$

where $P(\boldsymbol{X_r}|\boldsymbol{U_r})$ equals 1 if the codeword $\boldsymbol{X_r}$ corresponds to the information bits sequence $\boldsymbol{U_r}$ and 0 otherwise. For a convolutional code, this factor expands in the same way as for a SISO decoder (see section 2.2.2). In order to obtain the second factor of (5.7) and to factorize it further we need to make the assumption that the bits $U_{r,k}$ are independent given $\boldsymbol{Y_{sr}}$:

$$P(\boldsymbol{U_r}|\boldsymbol{y_{sr}}) = \prod_{k=1}^{K} P(U_{r,k}|\boldsymbol{y_{sr}}), \tag{5.8}$$

where the $P(U_{r,k}|\boldsymbol{y_{sr}})$ are computed by the SISO decoder. This independence assumption amounts to neglecting the correlation introduced by the SISO decoder between elements of $\boldsymbol{U_r}$.

For a convolutional code, the corresponding factor graph has strong similarities with the factor graph of the BCJR decoder (fig. 2.4 and 2.5), because of the formal similarities between (5.7)-(5.8) and the factorization of section 2.2.2 (when assuming BCJR modulation, i.e., $\boldsymbol{S} = \boldsymbol{X}$, and exchanging the roles of information bits and parity bits).

As a consequence, the application of the sum-product algorithm to this factor graph gives slightly modified versions of the standard forward recursion $\alpha(\cdot)$ and reverse recursion $\beta(\cdot)$ of the BCJR decoding algorithm. For an encoder of state $\boldsymbol{T_k}$ at time k, we can write the following recursions where each term in the summations has to

correspond to a combination of $(\boldsymbol{T_k}, U_{r,k}, X_{r,k}, \boldsymbol{T_{k+1}})$ satisfying the code constraints :

$$\begin{aligned}\alpha_{k+1}(\boldsymbol{T_{k+1}}) &= \sum_{\sim \boldsymbol{T_{k+1}}} \alpha_k(\boldsymbol{T_k}) \cdot P(U_{r,k}|\boldsymbol{y_{sr}}),\\ \beta_k(\boldsymbol{T_k}) &= \sum_{\sim \boldsymbol{T_k}} \beta_{k+1}(\boldsymbol{T_{k+1}}) \cdot P(U_{r,k}|\boldsymbol{y_{sr}}).\end{aligned}$$

We finally compute

$$P(X_{r,k}|\boldsymbol{y_{sr}}) = \sum_{\sim X_{r,k}} \alpha_k(\boldsymbol{T_k})\beta_{k+1}(\boldsymbol{T_{k+1}})P(U_{r,k}|\boldsymbol{y_{sr}}).$$

5.5.2 Equivalent Channel Model

We need to re-transmit the signal lying behind the probabilities $P(X_{r,k}|\boldsymbol{y_{sr}})$. Since we have no analytical model for the output of the SISO encoding/decoding process, we deduce this signal from the log-ratios under the assumption that they have the same form as at the output of an equivalent AWGN channel with BPSK modulation :

$$L_{X_{r,n}} \triangleq \log \frac{P(X_{r,n}=1|\boldsymbol{y_{sr}})}{P(X_{r,n}=-1|\boldsymbol{y_{sr}})} = L_{eq}(X_{r,n} + Z_{eq,n}). \tag{5.9}$$

The amplitude factor L_{eq} reflects the signal reliability and $Z_{eq,n}$ is a zero-mean white Gaussian noise sample of variance σ^2_{eq}. Both amplitude L_{eq} and variance σ^2_{eq} of the noise depend on the signal-to-noise ratio inside a block on the inter-user channel. On fading channels, they change thus at each frame. Their relation with the noise variance on the inter-user channel has to be computed empirically.

From this interpretation of the log-ratios, the relay constructs and forwards to the destination the signal $\beta_r(X_{r,n} + Z_{eq,n})$ which contains the payload plus an additive Gaussian noise. The factor β_r is computed in order to normalize the power transmitted in the frame :

$$\beta_r = \frac{1}{L_{eq}}\sqrt{\frac{P_r}{1+\sigma^2_{eq}}}. \tag{5.10}$$

The signal transmitted by the relay is thus

$$\begin{aligned}S_{r,n} &= \beta_r \cdot L_{X_{r,n}} &(5.11)\\ &= \sqrt{\frac{P_r}{1+\sigma^2_{eq}}} \cdot (X_{r,n} + Z_{eq,n}) &(5.12)\end{aligned}$$

for $n = 1, \ldots, N$.

This model suggests to interpret the transmission of coded data on the inter-user channel followed by the soft decoding/re-encoding process as the transmission of coded data on a global super-channel with better SNR. This interpretation is depicted in figure 5.4.

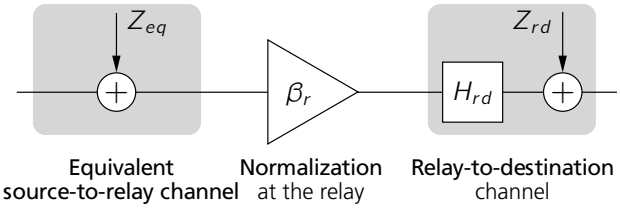


Figure 5.4: Equivalent user–relay–destination channel for Soft-DF.

The SNR enhancement is quantified in figure 5.5. This figure presents in its upper part the empirical input-output relation of the SISO decoder/re-encoder in terms of SNR, for the simulation parameters of section 5.5.4 and under the previous assumption of a gaussian i.i.d. noise corrupting the output. For a coded signal received with a SNR of 5 dB for example, the relay produces an output signal with a SNR of about 12 dB. The limitations of the i.i.d. Gaussian noise assumption appear distinctly below 0 dB

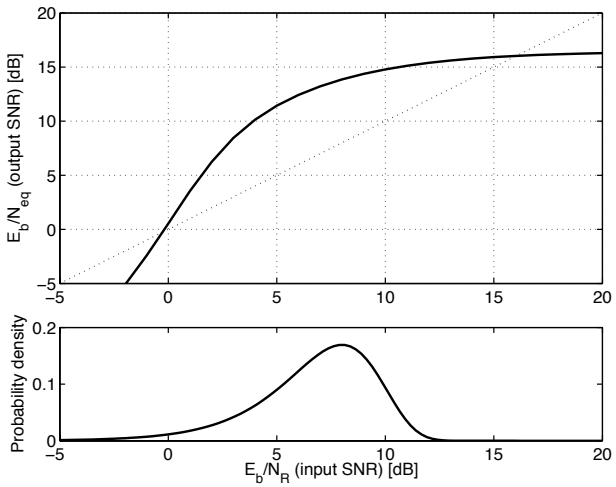


Figure 5.5: Input-output relation of the decoding/re-encoding process, in terms of instantaneous SNRs (i.e. for a particular realization of the fading).

and above 15 dB where the output SNR is lower than the input SNR. However for

common channel parameters, input SNRs lie mainly in the range of interest, as shown in the lower part of figure 5.5 by the probability density function of the input SNR for a Rayleigh fading channel with average SNR of 8 dB.

5.5.3 Reception and decoding

The destination receives the following signal from the relay :

$$\begin{aligned} \boldsymbol{Y_{rd}} &= H_{rd}\,\boldsymbol{S_r} + \boldsymbol{Z_{rd}} && (5.13)\\ &= H_{rd}\sqrt{\frac{P_r}{1+\sigma_{eq}^2}}\cdot\left(\boldsymbol{X_r} + \boldsymbol{Z_{eq}}\right) + \boldsymbol{Z_{rd}}, && (5.14) \end{aligned}$$

which is simply under the previous assumptions the output of a cascade composition of two channels corrupted by gaussian noise. Of course the receiver needs to estimate the parameters of the compound channel only and not the parameters of both channels separately, thus no channel state information for the individual channels is required.

The decoding process of this signal jointly with the one from the direct path is achieved by a standard turbo decoding operation.

5.5.4 Performance Comparison

Simulation results confirm the advantages of the Soft-DF relay technique over AF and DF through the following analysis of bit error rate (BER) curves. Figure 5.6 compares the performance of AF, DF and Soft-DF on Rayleigh block fading channels, with the following parameters. The source terminal encodes its data blocks of length $K = 1024$ bits with an RSC encoder of rate $1/2$, constraint length 4 and generator polynomials $(13_8, 17_8)$. The relay, receiving $2K$ symbols, forwards a block of another K symbols : the DF relay re-encodes the decoded bits with the same RSC code and forwards the parity bits only, the AF relay forwards the parity bits only, and the Soft-DF relay forwards the K parity bits computed with the non-recursive convolutional soft-encoder of constraint length 2 and generator polynomial 3_8. To decide whether to cooperate or not, the DF relay checks if a frame is error-free thanks to a 16-bit CRC code with generator polynomial given by the coefficients 0x1021 (hexadecimal notation). The bandwidth overhead required to transmit the 16 CRC bits is neglected. The figure compares the different schemes for equal overall energy consumption per information bit. The source terminal and the destination are at a normalized distance 1, and the relay is on the line joining them at a distance $d = 0.8$ from the source terminal (same

geometry as in fig. 3.4). The path loss exponent is chosen equal to 3.5. Denoting $E_{b,s}$ the energy spent by the source per information bit, the relay spends an additional average energy $E_{b,s}/2$ to cooperate. The turbo-decoders of DF and Soft-DF perform five iterations, which is enough to achieve convergence.

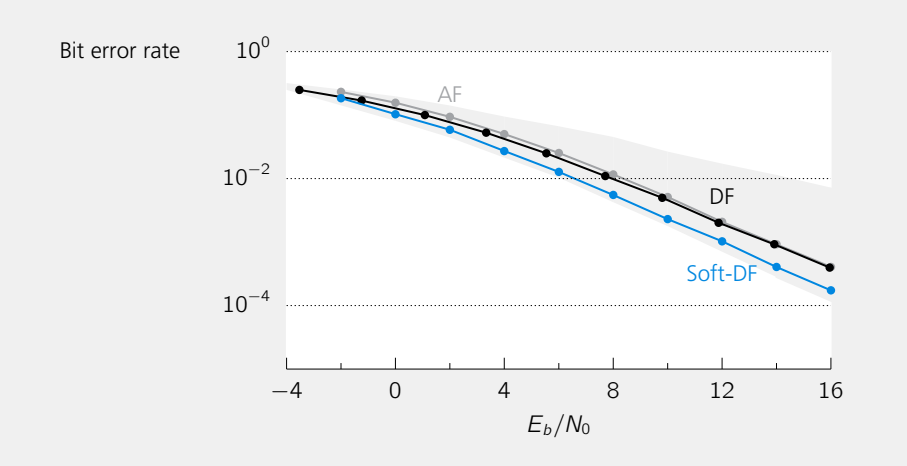


Figure 5.6: BER curves for AF, DF and Soft-DF on Rayleigh block fading channels. The filled area delimits the BERs achievable by allowing cooperation : the upper bound corresponds to non-cooperative transmission and the lower bound corresponds to transmission with perfect knowledge of $\boldsymbol{Y}_{sr}$ at the destination (equivalent to single-antenna to two-antennas transmission).

To achieve a BER of 10^{-3}, Soft-DF requires almost 2 dB less power than AF or DF. The figure also shows an upper bound and a lower bound on the achievable BER; these two bounds delimit the shaded area. The upper bound corresponds to non-cooperative transmission (turbo-code of rate $1/3$ with the same constituent codes as for DF, and an equal total energy expense), and the lower bound corresponds to transmission with perfect knowledge of $\boldsymbol{Y}_{sr}$ at the destination (equivalent to single-antenna to two-antennas transmission). Soft-DF almost achieves the BER of the lower bound.

These results reflect the advantages of the principles underlying Soft-DF : in comparison to DF, Soft-DF gains from an increased cooperation level and compared to AF, it wastes less power to transmit noise.

5.6 ERROR-RESILIENT RECEIVER FOR DECODE-FORWARD

In the presentation of the DF protocol of section 5.3, we have assumed that the relay forwards only codewords decoded without errors.

In this section instead, we consider a DF relay that forwards data even if it contains errors. We modify the MAP receiver at the destination to take into account the probability of errors on the source-to-relay link. The system is identical to the one of fig. 5.1, for code choices corresponding to the distributed turbo-coded DF scheme of section 5.3 (other code choices are possible).

The source terminal transmits information bits $\boldsymbol{U_s}$, encoded into coded bits $\boldsymbol{X_s}$ and modulated into symbols $\boldsymbol{S_s}$. The signals received by the relay and the destination are, respectively, $\boldsymbol{Y_{sr}}$ and $\boldsymbol{Y_{sd}}$.

The relay decodes the signal $\boldsymbol{Y_{sr}}$, makes hard decisions $\boldsymbol{U_r}$ and checks if errors occured thanks to a parity check code. Unlike in a DF scheme, the signal is forwarded whether it contains errors or not, after interleaving, re-encoding into $\boldsymbol{X_r}$ and modulation into $\boldsymbol{S_r}$. The relay also transmits to the destination some side information related to the source-to-relay channel state : if errors occured, it transmits the observed value $H_{sr} = h_{sr}$ as channel state information, so that the destination can determine the corresponding average error rate $P_e(h_{sr})$ in the relayed codeword; otherwise it declares that the signal was error-free. The destination then receives the signal $\boldsymbol{Y_{rd}}$ from the relay.

The destination uses both the direct and relayed signals $\boldsymbol{Y_{sd}}$ and $\boldsymbol{Y_{rd}}$ to make a joint decision. The decoding algorithm is presented in the following section.

5.6.1 Decoding algorithm

At the destination we are interested in finding the optimal estimates of the information bits in the maximum-a-posteriori (MAP) sense. For each bit $U_{s,k}$, where k is the index of the bit in the sequence $\boldsymbol{U_s}$, these estimates $\hat{U}_{s,k}$ are defined as follows :

$$\hat{U}_{s,k} \triangleq \arg \max_{U_{s,k}=-1,1} P(U_{s,k}|\boldsymbol{Y_{sd}}, \boldsymbol{Y_{rd}}). \tag{5.15}$$

The decoding algorithm is of course a standard turbo-decoding algorithm if no errors occur on the source-to-relay link, otherwise it needs to be modified.

The differences with the usual turbo-decoding algorithm are easily derived in the factor graphs framework. We will adopt this approach here : we first factorize the a

posteriori probability function in (5.15), then we draw the corresponding graph, and finally we apply the message-passing algorithm to it. We will focus our attention on the differences with the usual algorithm.

The a posteriori probability $P(U_{s,k}|\boldsymbol{Y_{sd}}, \boldsymbol{Y_{rd}})$ is the marginal function of the a posteriori probability of the whole sequence $P(\boldsymbol{U_s}|\boldsymbol{Y_{sd}}, \boldsymbol{Y_{rd}})$, and the maximization of the latter amounts to the maximization of the likelihood $P(\boldsymbol{Y_{sd}}, \boldsymbol{Y_{rd}}|\boldsymbol{U_s})$:

$$\hat{U}_{s,k} = \arg \max_{U_{s,k}=-1,1} \sum_{\sim\{U_{s,k}\}} P(\boldsymbol{U_s}|\boldsymbol{Y_{sd}}, \boldsymbol{Y_{rd}}) \tag{5.16}$$

$$= \arg \max_{U_{s,k}=-1,1} \sum_{\sim\{U_{s,k}\}} P(\boldsymbol{Y_{sd}}, \boldsymbol{Y_{rd}}|\boldsymbol{U_s}), \tag{5.17}$$

where $\sum_{\sim\{U_{s,k}\}}$ is the sum over all bits of $\boldsymbol{U_s}$ (and $\boldsymbol{U_r}$ in (5.18)-(5.19)) except $U_{s,k}$. In this expression we can make the dependence between $\boldsymbol{U_s}$ and $\boldsymbol{U_r}$ explicit, and then factorize it thanks to the independence of $\boldsymbol{Y_{sd}}$ and $\boldsymbol{Y_{rd}}$ given $\boldsymbol{U_s}$ and $\boldsymbol{U_r}$:

$$\hat{U}_{s,k} = \arg \max_{U_{s,k}=-1,1} \sum_{\sim\{U_{s,k}\}} P(\boldsymbol{Y_{sd}}, \boldsymbol{Y_{rd}}|\boldsymbol{U_s}, \boldsymbol{U_r}) \cdot P(\boldsymbol{U_r}|\boldsymbol{U_s}) \tag{5.18}$$

$$= \arg \max_{U_{s,k}=-1,1} \sum_{\sim\{U_{s,k}\}} P(\boldsymbol{Y_{sd}}|\boldsymbol{U_s}) \cdot P(\boldsymbol{Y_{rd}}|\boldsymbol{U_r}) \cdot P(\boldsymbol{U_r}|\boldsymbol{U_s}). \tag{5.19}$$

The factor $P(\boldsymbol{U_r}|\boldsymbol{U_s})$ reflects the uncertainty of $\boldsymbol{U_r}$ and is responsible for the differences with the usual turbo-decoding algorithm. Continuing this derivation in the factor graphs framework, the first two factors yield the convolutional decoders (as seen in section 2.2.2) and the third describes the exchanges of information between them. The exact description of the factor $P(\boldsymbol{U_r}|\boldsymbol{U_s})$ would require to model the errors in the sequence $\boldsymbol{U_r}$, resulting from its coded transmission through the source-to-relay channel. A more simple yet clearly suboptimal assumption is to consider the errors as independent, neglecting the correlation between errors at the output of the convolutional code decoder. We write this assumption as follows :

$$P(\boldsymbol{U_r}|\boldsymbol{U_s}) = \prod_k P(U_{r,k}|U_{s,k}), \tag{5.20}$$

where $P(U_{r,k}|U_{s,k})$ is considered as constant for all k. It takes the following values

$$P(U_{r,k}|U_{s,k}) = \begin{cases} 1 - P_e & \text{if } U_{r,k} = U_{s,k} \\ P_e & \text{if } U_{r,k} \neq U_{s,k}, \end{cases} \tag{5.21}$$

where P_e is the average error rate between the source and the relay. Note that this error rate depends on the instantaneous signal-to-noise ratio on the source-to-relay channel.

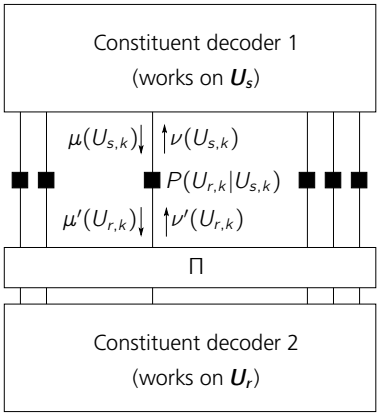


Figure 5.7: Outlines of the graph representing the iterative (turbo) decoding algorithm.

Given the factorizations (5.19) and (5.20), the factor graph is drawn in figure 5.7. The nodes corresponding to the factors (5.20) appear between the two subgraphs corresponding to the constituent decoders.

From the factor graph, deriving the modification of the usual algorithm is now straightforward. The information messages (shown in figure 5.7) sent by a constituent decoder are modified before they reach the other decoder, according to the following update equations :

$$\begin{aligned} \mu'_k(U_{r,k}) &= \sum_{U_{s,k}=-1,1} P(U_{r,k}|U_{s,k})\,\mu_k(U_{s,k}) \\ \nu_k(U_{s,k}) &= \sum_{U_{r,k}=-1,1} P(U_{r,k}|U_{s,k})\,\nu'_k(U_{r,k}), \end{aligned} \tag{5.22}$$

for all k. This achieves the translation of information available on $U_{s,k}$ to information on $U_{r,k}$ and conversely. Besides this modification, the other difference with the usual turbo-decoding algorithm is that the second constituent decoder works on the bits $\boldsymbol{U_r}$.

5.6.2 Validation of the algorithm

In this section the transmission scheme and decoding algorithm are validated by simulations. Their performance in terms of bit error rate (BER) is shown to be superior to

that of the classical AF and DF cooperative schemes.

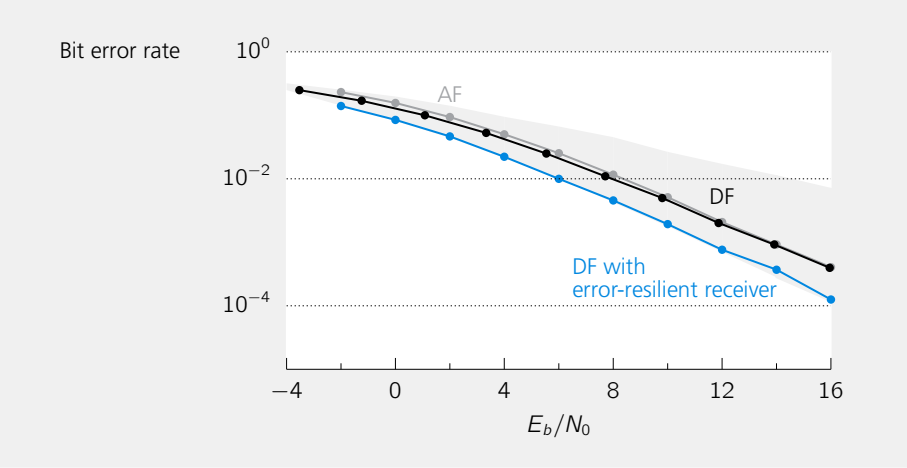


Figure 5.8: BER curves for the three cooperative strategies. The new DF strategy and the corresponding algorithm offer an improvement of 2 dB or more, for BERs below 10^{-3}. The filled area delimits the BERs achievable by allowing cooperation : the upper bound corresponds to non-cooperative transmission and the lower bound corresponds to transmission with perfect knowledge of $\boldsymbol{Y}_{sr}$ at the destination (equivalent to single-antenna to two-antennas transmission).

Figure 5.8 presents BER curves for the three different transmission strategies, with the following parameters. The source encodes its data blocks of length $K = 1024$ bits with a recursive systematic convolutional (RSC) encoder of rate $1/2$, constraint length 5 and generator polynomials $(23_8, 37_8)$ (octal notation). For the DF schemes, the relay decodes using a BCJR decoder, and re-encodes with the same RSC code as at the source. The interleaver in between is chosen randomly. Error detection as the relay is considered as optimal and the few additional parity check bits needed for error detection are neglected. Channel coefficients H_{sr}, H_{rd} and H_{sd} incorporate both Rayleigh fading and path-loss depending on the relative positions of the devices. The path loss exponent is chosen equal to 3.5 and the distance between the source and the destination is normalized to 1. The relay is located on the line joining the source and the destination at a distance 0.8 from the source. BER curves are plotted as a function of E_b/N_0, where E_b is the total energy per bit spent by the source and the relay all together.

The bit error rate achieved with the new decoding algorithm is significantly lower in comparison with the usual AF and DF strategies. For bit error rates of 10^{-3} and lower, the gain is at least 2 dB; in fact the BER of the new scheme is barely distinguishable, on this figure, from the BER achieved with perfect knowledge of $\boldsymbol{Y}_{sr}$ at the destination

(lower limit of the filled area).

In this setup, where the relay is rather far from the source, the usual DF strategy is limited by the low received power at the relay that often prevents successful decoding and forwarding. The new algorithm presented here however, is able to use the erroneous frames forwarded by the relay to enhance the final BER, as figure 5.8 demonstrates.

5.6.3 Comments on the information exchanges

Some insight on the behavior of the algorithm can be gained by analyzing the exchanges of messages between the two constituent decoders. These exchanges are limited by the transformations (5.22) because their outputs always satisfy

$$\begin{aligned} \mu'_k(U_{r,k}) &\leq 1 - P_e \\ \nu_k(U_{s,k}) &\leq 1 - P_e, \end{aligned} \tag{5.23}$$

which is a straightforward consequence of (5.22) since $\mu_k(-1)+\mu_k(1) = 1$, $\nu'_k(-1)+\nu'_k(1) = 1$, and the functions μ_k, ν'_k are always positive. This reflects that even the certitude on the value of the bit $U_{r,k}$ only enables to determine the value of the bit $U_{s,k}$ with a probability $1 - P_e$, since there is still a probability P_e that these two bits differ. No further improvements are possible when the bounds (5.23) are reached.

The evolution of the information exchanges can be presented in terms of mutual information. We are interested in the information available on $U_{s,k}$ at the input and at the output of the first constituent decoder, in the $\mu_k(U_{s,k})$ and $\nu_k(U_{s,k})$ messages. The corresponding mutual informations are written $I(U_{s,k}; \mu_k(U_{s,k}))$ and $I(U_{s,k}; \nu_k(U_{s,k}))$. The improvements of these mutual informations throughout the iterations of the decoding algorithm can be computed by simulation, and then displayed as a trajectory much like the trajectory of mutual information in an EXIT chart. Figure 5.9 shows three such trajectories, for three values of the instantaneous E_b/N_0 and the following parameters : RSC code with polynomials (23_8, 37_8), probability of error at the relay fixed to 0.01, fixed channel coefficients $H_{sd} = H_{rd} = 1$ and relay located at a normalized distance 0.2 from the source.

As it can be expected from (5.23), the incertitude between $\boldsymbol{U_s}$ and $\boldsymbol{U_r}$ limits the mutual information exchanged between the codes. In figure 5.9, the two trajectories with the highest values of E_b/N_0 increase until $I(U_{s,k}; \nu_k(U_{s,k}))$ reaches its maximal achievable value (vertical dotted line on the right). This maximal mutual information is reached

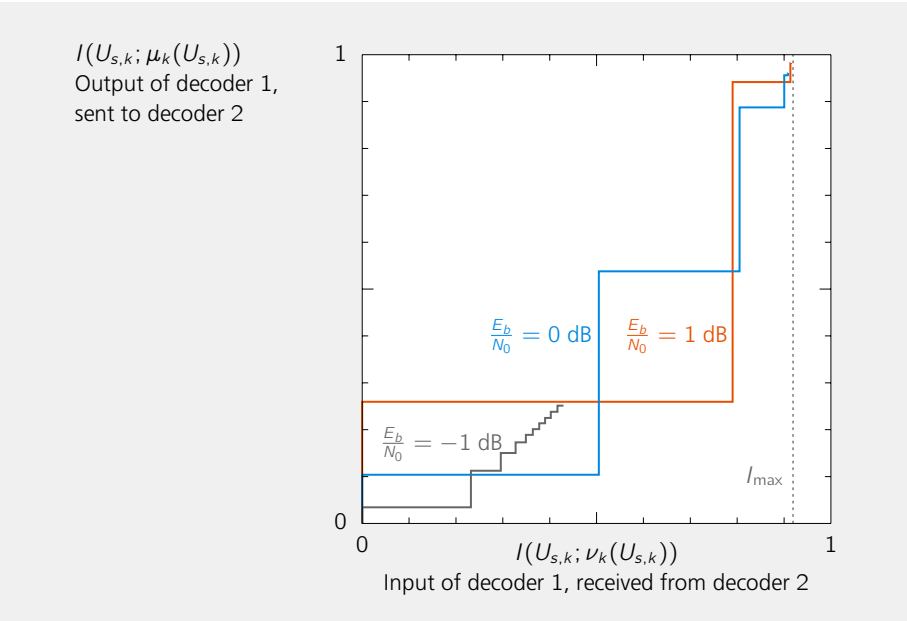


Figure 5.9: Evolution of the mutual informations $I(U_{s,k};\mu_k(U_{s,k}))$ and $I(U_{s,k};\nu_k(U_{s,k}))$ during the decoding of a sequence, for three instantaneous E_b/N_0.

when $U_{r,k}$ is known with certitude; in that case we have either

$$\begin{cases} \nu_k(1) = 1 - P_e \\ \nu_k(-1) = P_e \end{cases} \text{ if } U_{r,k} = 1, \text{ or } \begin{cases} \nu_k(1) = P_e \\ \nu_k(-1) = 1 - P_e \end{cases} \tag{5.24}$$

if $U_{r,k} = -1$; the corresponding value of $I(U_{s,k};\nu_k(U_{s,k}))$ is

$$I_{\max} = 1 + P_e \log_2(P_e) + (1 - P_e) \log_2(1 - P_e). \tag{5.25}$$

In the reverse sense, the mutual information $I(U_{r,k};\mu'_k(U_{r,k}))$ going from the first to the second constituent decoder is also bounded by $I_{\max}$.

5.7 ITERATIVE RECEIVER FOR COMPRESS-FORWARD

This section addresses the CF protocol, and proposes a new receiver for decoding in the MAP sense at the destination.

As described in section 3.5, the CF protocol uses compression with side-information, i.e., Wyner-Ziv coding [55]; the side-information available at the destination is the

signal received on the direct user-to-destination link.

A practical way to implement Wyner-Ziv coding is to use scalar quantization followed by Slepian-Wolf source coding [79], as suggested in [80,81]. Several coded implementations of the CF protocol do so, using scalar quantization at the relay [74], followed by source coding [69,75] or Slepian-Wolf source coding [76].

While several such code and quantizer implementations have already been proposed, the corresponding decoder design as received comparatively less attention. Up to now, receivers used at the destination have followed the lines of the CF decoder used with random codes in [38]. These receivers decode the channel code used by the relay, then decode the Slepian-Wolf source code, reconstruct the quantization samples and finally, use them for decoding the signal sent by the source terminal.

We were interested instead in the maximum-a-posteriori (MAP) receiver for the CF protocol, and in its consequences on the design of the system.

This section presents the derivation of the MAP receiver in the factor graphs framework, for a half-duplex orthogonal relay channel. The MAP CF decoder is found to be iterative : it alternates between decoding the signals from the user terminal and from the relay terminal. Improvements made by each constituent decoder are recycled by the other, as respectively better quantization estimates or better side information. Exchanges are made of so-called soft information : the dequantizer for example, does never choose a definite reconstruction value for the quantized samples; instead it keeps probability information on possible reconstruction values. Differing from other algorithms, the side information is also refined during the iterative process; these refinements are made possible by using the code underlying the signal of side information. The performance of this algorithm is evaluated by simulations, for a simple CF system with scalar quantization and Slepian-Wolf source coding.

The analysis of the receiver also yields a design criterion for the scalar quantizer, because the latter determines how information is exchanged between the two constituent decoders. A natural tool for the analysis of the quantizer are the extrinsic information transfer functions (EXIT functions) of these exchanges. The quantizer can thus be chosen so as to yield the most appropriate EXIT functions. As a consequence, besides improving decoding performance, using the MAP receiver also impacts the system design.

In addition to the MAP receiver design, this section ends with a comparison between relays quantizing channel observations or soft decoder outputs (as in section 5.6).

Although the quantization of channel observations was proposed in [38], several works use relays that quantize the outputs of a soft decoder, e.g. [7] (see section 5.6) or [75, 74]. The idea of preceding the quantizer with a soft channel decoder is to improve the knowledge of the information bits of the user by using the channel code; this can be thought of as a kind of vector pre-processing before scalar quantization. In our case however, results show that quantizing channel observations works better, and suggest that errors made by the soft-decoder tend to propagate to the destination.

The section is organized as follows. Section 5.7.1 introduces the system model and section 5.7.2 presents the derivation of the MAP CF decoding algorithm. The receiver is then validated by simulations in section 5.7.3. Section 5.7.4 analyzes the dequantizer and identifies its EXIT functions as a design criterion. Finally, section 5.7.5 compares the quantization of channel observations or soft decoder outputs.

5.7.1 System model

We consider the system of fig. 5.10.

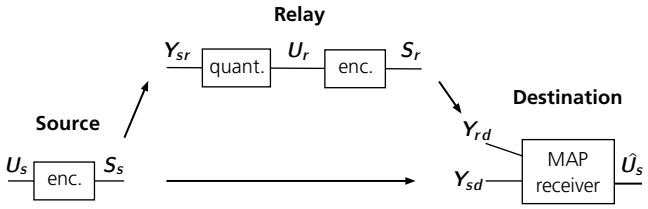


Figure 5.10: Cooperative system. The abbreviations *enc.* and *quant.* stand for *encoder* and *quantizer*.

The user transmits a sequence of information bits $\boldsymbol{U_s}$ (length K), coded and modulated into a sequence of symbols $\boldsymbol{S_s}$ (length N). From the transmission of these symbols the relay receives $\boldsymbol{Y_{sr}}$ and the destination $\boldsymbol{Y_{sd}}$.

The relay quantizes $\boldsymbol{Y_{sr}}$, and after coding and modulation, forwards a signal $\boldsymbol{S_r}$ to the destination. This sequence is received as $\boldsymbol{Y_{rd}}$ by the destination.

Finally, the destination uses both the direct and relayed signals $\boldsymbol{Y_{sd}}$ and $\boldsymbol{Y_{rd}}$ to make a joint decision. The decoding algorithm is derived in section 5.7.2.

Processing at the relay

The compression takes advantage of the side information available at the destination : since $\boldsymbol{Y_{sr}}$ is correlated to $\boldsymbol{Y_{sd}}$, available at the destination, the relay uses Wyner-Ziv coding [55]. In our system, Wyner-Ziv coding is achieved by scalar quantization followed by Slepian-Wolf source coding, as suggested in [80, 81], or specifically for the relay channel in [76].

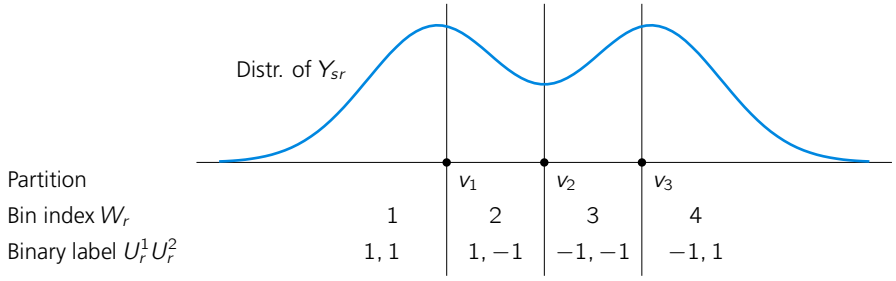


Figure 5.11: The quantizer (here with 4 bins) is defined by its partition in bins and by the binary labeling of the bins.

The scalar quantizer of the relay is defined by its partition in bins and by the binary labeling of the bins (example in fig. 5.11). The partition consists of B bins $W = 1, \ldots, B$, delimited by $B - 1$ thresholds $v_1, \ldots, v_{B-1}$; it is symmetric around the origin because the signal to quantize has a symmetric distribution. The labeling function $\mathcal{L}(\cdot)$ defines a correspondence between a bin index W and a unique binary label $U_r^1, \ldots, U_r^Q$ of length Q :

$$[U_r^1, \ldots, U_r^Q] = \mathcal{L}(W). \tag{5.26}$$

After the scalar quantization, the sequence $\boldsymbol{U_r}$ of binary labels is encoded by a Slepian-Wolf source code, a channel code and is modulated into a sequence of symbols $\boldsymbol{S_r}$.

5.7.2 Iterative receiver

The Compress-Forward protocol for random codes in [38] uses at the destination a receiver that first decodes the bits $\boldsymbol{U_r}$, then makes estimates of $\boldsymbol{Y_{sr}}$ using the signal from the user $\boldsymbol{Y_{sd}}$ as side information, and finally decodes the information $\boldsymbol{U_s}$. Practical implementations of Compress-Forward [75, 69, 74, 76] use this same sequence of operations.

This section addresses instead decoding according to the maximum-a-posteriori (MAP) criterion. The MAP estimates of the user's bits U_k ($k = 1, \ldots, K$) are defined as follows :

$$\hat{U}_{s,k} \triangleq \arg \max_{U_{s,k}=-1,1} P(U_{s,k} | \boldsymbol{Y_{sd}} = \boldsymbol{y_{sd}}, \boldsymbol{Y_{rd}} = \boldsymbol{y_{rd}}). \tag{5.27}$$

A decoding algorithm for this problem is easily derived in the factor graphs and sum-product algorithm framework.

The resulting algorithm differs from those found in the literature in three ways. First, it decodes $\boldsymbol{Y_{sd}}$ and $\boldsymbol{Y_{rd}}$ jointly, by iterating between the corresponding constituent decoders. Second, rather than taking a decision on reconstructed values for the compressed samples of $\boldsymbol{Y_{sr}}$, it weights possible alternatives by their respective probability. Third and last, the decompression uses the improving knowledge of $\boldsymbol{S_s}$ along iterations to better exploit the correlation between $\boldsymbol{Y_{sr}}$ and the side information $\boldsymbol{Y_{sd}}$.

Factor graph for the whole receiver

This section addresses the factorization of (5.27) and presents the corresponding factor graph.

The a posteriori probability $P(U_{s,k} | \boldsymbol{y_{sd}}, \boldsymbol{y_{rd}})$ is the marginal function of the a posteriori probability of the whole sequence $P(\boldsymbol{U_s} | \boldsymbol{y_{sd}}, \boldsymbol{y_{rd}})$, and the maximization of the latter amounts to the maximization of the likelihood $P(\boldsymbol{y_{sd}}, \boldsymbol{y_{rd}} | \boldsymbol{U_s})$ for equiprobable bits $\boldsymbol{U_s}$:

$$\begin{aligned} \hat{U}_{s,k} &= \arg \max_{U_{s,k}=-1,1} \sum_{\sim\{U_{s,k}\}} P(\boldsymbol{U_s} | \boldsymbol{y_{sd}}, \boldsymbol{y_{rd}}) && (5.28) \\ &= \arg \max_{U_{s,k}=-1,1} \sum_{\sim\{U_{s,k}\}} P(\boldsymbol{y_{sd}}, \boldsymbol{y_{rd}} | \boldsymbol{U_s}). && (5.29) \end{aligned}$$

In this expression we can make the dependence between $\boldsymbol{U_s}$ and $\boldsymbol{U_r}$ explicit, and then factorize thanks to the independence of $\boldsymbol{Y_{sd}}$ and $\boldsymbol{Y_{rd}}$ given $\boldsymbol{U_s}$ and $\boldsymbol{U_r}$:

$$\begin{aligned} \hat{U}_{s,k} &= \arg \max_{U_{s,k}=-1,1} \sum_{\sim\{U_{s,k}\}} P(\boldsymbol{y_{sd}}, \boldsymbol{y_{rd}} | \boldsymbol{U_s}, \boldsymbol{U_r}) \cdot P(\boldsymbol{U_r} | \boldsymbol{U_s}) && (5.30) \\ &= \arg \max_{U_{s,k}=-1,1} \sum_{\sim\{U_{s,k}\}} P(\boldsymbol{y_{sd}} | \boldsymbol{U_s}) \cdot P(\boldsymbol{y_{rd}} | \boldsymbol{U_r}) \cdot P(\boldsymbol{U_r} | \boldsymbol{U_s}). && (5.31) \end{aligned}$$

Noting that $\boldsymbol{U_s} - \boldsymbol{S_s} - \boldsymbol{Y_{sd}}$ and $\boldsymbol{U_r} - \boldsymbol{S_r} - \boldsymbol{Y_{rd}}$ are Markov chains, the first two factors

of (5.31) can be factorized as follows :

$$P(\boldsymbol{y_{sd}}|\boldsymbol{U_s}) = P(\boldsymbol{y_{sd}}|\boldsymbol{S_s}) \cdot P(\boldsymbol{S_s}|\boldsymbol{U_s}), \tag{5.32}$$

$$P(\boldsymbol{y_{rd}}|\boldsymbol{U_r}) = P(\boldsymbol{y_{rd}}|\boldsymbol{S_r}) \cdot P(\boldsymbol{S_r}|\boldsymbol{U_r}). \tag{5.33}$$

The third factor, $P(\boldsymbol{U_r}|\boldsymbol{U_s})$, reflects the correlation between the information bits $\boldsymbol{U_s}$ and the sequence $\boldsymbol{U_r}$ sent by the relay. Since $\boldsymbol{U_s} - \boldsymbol{S_s} - \boldsymbol{W_r} - \boldsymbol{U_r}$ is a Markov chain, we have

$$P(\boldsymbol{U_r}|\boldsymbol{U_s}) = P(\boldsymbol{U_r}|\boldsymbol{W_r}) \cdot P(\boldsymbol{W_r}|\boldsymbol{S_s}) \cdot P(\boldsymbol{S_s}|\boldsymbol{U_s}). \tag{5.34}$$

The factorization (5.31) can thus be written :

$$\hat{U}_{s,k} = \arg \max_{U_{s,k}=-1,1} \sum_{\sim\{U_{s,k}\}} P(\boldsymbol{y_{sd}}|\boldsymbol{S_s}) \cdot P^2(\boldsymbol{S_s}|\boldsymbol{U_s}) \tag{5.35}$$

$$\cdot \; P(\boldsymbol{y_{rd}}|\boldsymbol{S_r}) \cdot P(\boldsymbol{S_r}|\boldsymbol{U_r}) \tag{5.36}$$

$$\cdot \; P(\boldsymbol{U_r}|\boldsymbol{W_r}) \cdot P(\boldsymbol{W_r}|\boldsymbol{S_s}). \tag{5.37}$$

Functional blocks of the decoding algorithm become apparent in this last factorization. Indeed, continuing the factorization in the factor graph framework,

- the factors in (5.35) yield the constituent decoder and demodulator for the signal $\boldsymbol{U_s}$ from the user (Note that the square can be omitted since $P(\boldsymbol{S_s}|\boldsymbol{U_s})$ is equal either to 0 or 1, because the correspondance between the sequence of information bits $\boldsymbol{U_s}$ and the sequence of symbols $\boldsymbol{S_s}$ is deterministic);
- the factors in (5.36) yield the constituent decoder and demodulator for the signal $\boldsymbol{U_r}$ from the relay;
- and finally, the factors in (5.37) describe the dequantization and, since it contains both $\boldsymbol{S_s}$ and $\boldsymbol{U_r}$, connects the former two constituent decoders.

Decoders for the constituent codes of the user and the relay are assumed to be known and are not addressed here. In the following we focus on the dequantization part.

We continue the factorization of (5.37) into symbol-level factors. Since the quantizer is scalar and memoryless, we can write

$$P(\boldsymbol{U_r}|\boldsymbol{W_r}) \cdot P(\boldsymbol{W_r}|\boldsymbol{S_s}) = \prod_{n=1}^{N} P(U_{r,n}^1 \ldots U_{r,n}^Q|W_{r,n}) \cdot P(W_{r,n}|S_{s,n}). \tag{5.38}$$

The factors $P(U_{r,n}^1 \dots U_{r,n}^Q | W_{r,n})$ describe the deterministic correspondence between a binary label $U_{r,n}^1 \dots U_{r,n}^Q$ and the quantizer bin number $W_{r,n}$, thus

$$P(U_{r,n}^1 \dots U_{r,n}^Q | W_{r,n}) = \begin{cases} 1 & \text{if } [U_{r,n}^1 \dots U_{r,n}^Q] = \mathcal{L}(W_{r,n}), \\ 0 & \text{otherwise.} \end{cases} \tag{5.39}$$

The factors $P(W_{r,n}|S_{s,n})$ are the probabilities that, when the user sends a symbol $S_{s,n}$, the relay receives a $Y_{sr,n}$ in the bin $W_{r,n}$ of the quantizer :

$$P(W_{r,n}|S_{s,n}) = \int_{v_{w-1}}^{v_w} \frac{1}{\sqrt{\pi N_0}} e^{-(y - H_{sr} S_{s,n})^2 / N_0} dy \tag{5.40}$$

if one defines $v_0 = -\infty$ and $v_B = +\infty$. This depends on the modulation used by the user, on the signal-to-noise ratio on the user-to-relay channel, and on the position of the bin $W_{r,n}$.

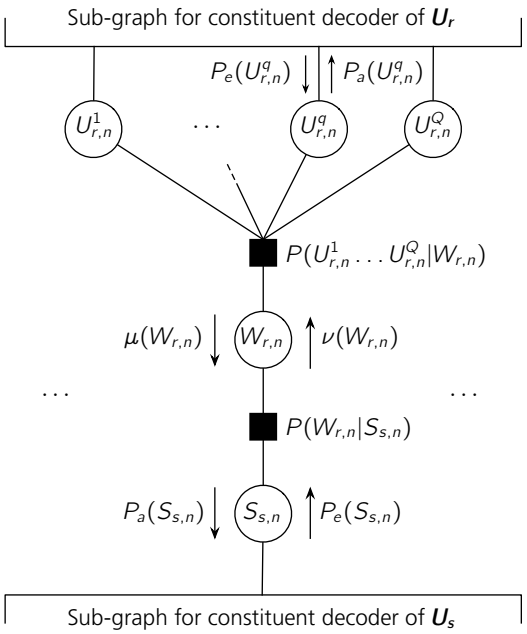


Figure 5.12: Outlines of the graph representing the iterative decoding algorithm. The arrows and their labels represent messages of the sum-product algorithm.

Given the factorizations (5.35)–(5.38), the factor graph is drawn in fig. 5.12. The

nodes corresponding to the dequantizer appear between the two sub-graphs corresponding to the constituent demodulators and decoders of $\boldsymbol{U_s}$ and $\boldsymbol{U_r}$.

Message-passing for the dequantizer

This section details the operations of the dequantizer, derived by applying the sum-product algorithm to the graph of fig. 5.12.

The sum-product algorithm applied to the dequantizer describes how extrinsic probabilities $P_e(U_{r,n}^q), q = 1, \ldots, Q$ and $P_e(S_{s,n})$ produced by the constituent decoders, are transformed into a priori probabilities $P_a(S_{s,n})$ and $P_a(U_{r,n}^q), q = 1, \ldots, Q$, for the other decoder.

In the relay-to-user decoder direction, the transformation is made of the following two steps :

$$\begin{aligned} \mu(W_{r,n}) &= P_e(U_{r,n}^1) \cdots P_e(U_{r,n}^Q) \quad \text{for } [U_{r,n}^1 \cdots U_{r,n}^Q] = \mathcal{L}(W_{r,n}), \quad (5.41) \\ P_a(S_{s,n}) &= \sum_{W_{r,n}=1}^{B} P(W_{r,n}|S_{s,n}) \cdot \mu(W_{r,n}). \quad (5.42) \end{aligned}$$

The dequantizer forms a priori information $P_a(S_{s,n})$ for the decoder of $\boldsymbol{U_s}$ from a weighted combination of all possible $W_{r,n}$, instead of choosing a definite reconstruction value.

Similarly, we have in the user-to-relay decoder direction :

$$\begin{aligned} \nu(W_{r,n}) &= \sum_{W_{r,n}=1}^{B} P(W_{r,n}|S_{s,n}) \cdot P_e(S_{s,n}), \quad (5.43) \\ P_a(U_{r,n}^q) &= \sum_{W_{r,n}:U_{r,n}^q} \nu(W_{r,n}) \prod_{\substack{i=1 \\ i \neq q}}^{Q} P_e(U_{r,n}^i), \quad (5.44) \end{aligned}$$

for $q = 1, \ldots, Q$, and where the notation $W_{r,n} : U_{r,n}^q$ indicates that the sum is over all values of $W_{r,n}$ compatible with $U_{r,n}^q$ through $\mathcal{L}(W_{r,n}) = [U_{r,n}^1, \ldots, U_{r,n}^q, \ldots, U_{r,n}^Q]$. The a priori probabilities $P_a(U_{r,n}^q)$ calculated by the dequantizer from the extrinsic information $P_e(S_{s,n})$ are used by the decoder of $\boldsymbol{U_r}$ as side information; they enable Slepian-Wolf source decoding.

The said dequantizer has thus two functions, one being the actual dequantization by (5.41)–(5.42) and the other being updating the side information by (5.43)–(5.44). By simplicity we keep the name dequantizer for both in the sequel, as they are a consequence of the presence of a quantizer.

Comments on the algorithm

In the previous two sections the MAP algorithm has been recognized to include constituent decoders for the signals $\boldsymbol{U_s}$ and $\boldsymbol{U_r}$, as well as a dequantizer that governs exchanges between them. We now highlight some features of the overall decoding process.

First, the presence of cycles in the factor graph makes the algorithm iterative. The constituent decoders will exchange information, through the dequantizer, in order to progressively refine their estimates. This is different from usual decoders that first decode $\boldsymbol{U_r}$ with $\boldsymbol{S_s}$ as side information, then decode $\boldsymbol{U_s}$, and do not iterate. Various scheduling between decoding operations are possible (e.g. start by decoding $\boldsymbol{U_s}$, or $\boldsymbol{U_r}$, etc.).

Second, the dequantizer never takes a definite decision on the estimates; instead it assigns probabilities to each possibility. For this reason, and following usual naming conventions, we will term this *soft dequantization*.

Third, the decoder would not be iterative if the data sent from the user was memoryless, because there would be no cycles (at least cycles including both the user and relay signals, creating iterations between the constituent decoders of $\boldsymbol{U_s}$ and $\boldsymbol{U_r}$). The receivers makes thus specifically use of the correlations between symbols $\boldsymbol{S_s}$ induced by the code. This results in refinements of $P_e(S_{s,n})$ over iterations, and by (5.43)–(5.44) in refinements of the side information for Slepian-Wolf decoding.

5.7.3 Validation of the algorithm

This section illustrates the performance improvements brought about by the joint MAP receiver. The following two sections present the codes and modulations used for the simulations, followed by some simulation results.

Choice of coding : a CF counterpart to Valenti's DF

For these simulations we use a CF coding scheme that can be thought of as a counterpart to the distributed turbo-coded DF scheme of Valenti et al. [63].

As described in section 5.3, the distributed turbo-coded DF scheme of Valenti consists in using convolutional codes at the user transmitter and at the relay in such a way that they make up a turbo-code. The advantage of this DF scheme is that it makes up a turbo-coded transmission, while keeping the complexity low at the relay since the relay only decodes a convolutional code, not a turbo-code.

The CF system used here is a counterpart to Valenti's DF scheme in the sense, first, that it also uses the same combination of convolutional codes and interleaver, and second, that it also aims at keeping the complexity low at the relay. This CF scheme is intended to enable the relay to switch from DF to CF when it does not successfully decode the signal it receives. Using the same codes for both schemes enables reusing the same hardware.

The transmitters and receivers are depicted in fig. 5.13. The user encodes its data with a recursive convolutional code and binary phase-shift keying (BPSK) modulation. The relay quantizes its channel observations with a scalar quantizer, and the resulting sequence of binary labels is interleaved. Interleaving makes the binary sequence appear uncorrelated for subsequent coding, and from a decoder perspective, avoids short cycles in the factor graph, that are known to degrade the performance of an iterative receiver. Interleaved bits are then encoded with a recursive convolutional code that acts as a source-channel code (for the use of recursive convolutional codes as source-channel codes, see [82, 83]). Parity bits produced by such a code are equiprobable, as shown in [84]. This sequence of parity bits is then modulated either with BPSK or, for higher rates, with M-size amplitude-shift keying (M-ASK). In the latter case, the binary sequence is interleaved before transmission in order to yield bit-interleaved coded modulation (BICM, see section 2.4).

Alternatively, the relay could encode into separate layers each position $q = 1, \ldots, Q$ of the binary labels output by the quantizer, since they have in general different statistics. As other alternatives, code choices in the literature include LDPC codes for the user in [69,76], and irregular-repeat-accumulate codes for Slepian-Wolf source-channel coding in [76].

As a consequence of our coding choices, the constituent decoders for $\boldsymbol{U_s}$ and $\boldsymbol{U_r}$ are BCJR decoders. The constituent decoder of $\boldsymbol{U_r}$ is complemented by an interleaver

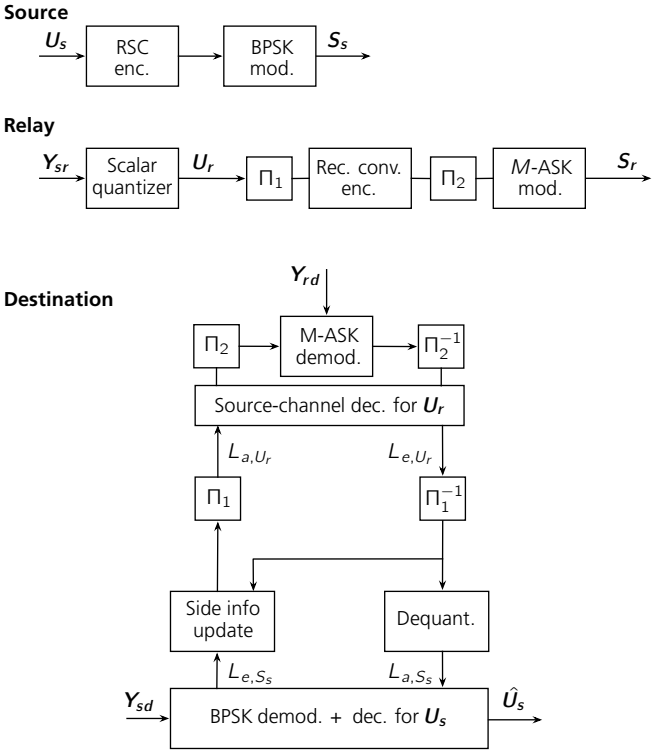


Figure 5.13: System used for simulations.

and an M-ASK demodulator to handle the BICM. The decoding algorithms are implemented in the log-domain, i.e. with probabilities represented by their log-ratio defined as follows :

$$L_{a,S_{s,n}} = \ln \frac{P_a(S_{s,n} = +1)}{P_a(S_{s,n} = -1)} \tag{5.45}$$

and similarly for $L_{e,S_{s,n}}$, $L_{a,U^q_{r,n}}$ and $L_{e,U^q_{r,n}}$.

The scheduling is as follows : the probabilities $P(y_{sd,n}|S_{s,n})$ initialize the extrinsic probabilities $P_e(S_{s,n})$, then the decoding operations are carried out as shown by the direction of the arrows in fig. 5.13. Such a scheduling makes the first iteration correspond to the decoders found in the literature. After several iterations the receiver finally outputs bit estimates $\hat{U}_{s,k}$.

Simulations

In this section the iterative decoding algorithm is validated by simulations. The iterative refinements are shown to improve significantly the performance in terms of bit error rate (BER).

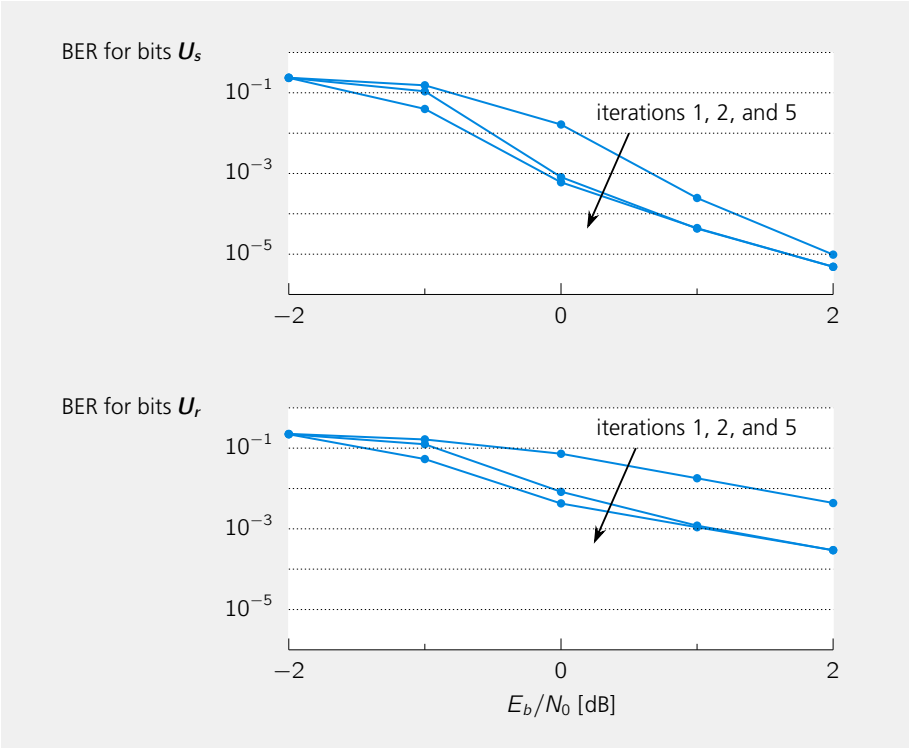


Figure 5.14: Bit error rate over iterations. The first iteration corresponds to decoders that follow the steps of the CF decoder for random codes.

Figure 5.14 presents curves of the BER for bits of $\boldsymbol{U_s}$ and $\boldsymbol{U_r}$, for increasing number of iterations, with the following parameters. The user encodes its data blocks of length $K = 1024$ bits with a recursive systematic convolutional encoder of rate $1/2$, constraint length 5 and generator polynomials $(23_8, 37_8)$ (octal notation). At the relay, the scalar quantizer has a partition of 4 bins, chosen so as to maximize the mutual information $I(S_s; W_r)$ (see section 5.7.4). The binary labeling of the bins is the same as in fig. 5.11. The interleaver is random and re-encoding is achieved with the same recursive code as at the source (systematic bits are not sent though, as in [84]). The relay uses 16-ASK modulation with set-partitioning mapping. The number of symbols sent by the relay is thus chosen to be half the number of symbols sent by the source.

All three terminals are lined up, the relay is between the user and the destination at a normalized distance $d = 0.7$ from the user. Path losses between the terminals determine the power levels at the receivers. The path loss exponent is chosen equal to 3.5. The two-sided power spectral densities of noise $N_0/2$ are set equal for both the relay and destination receivers. BER curves are plotted as a function of E_b/N_0, where E_b is the total energy per bit spent by the source and the relay all together. Within each global iteration, the BICM decoder for $\boldsymbol{U_r}$ iterates as long as it provides improvements; in practice 4 iterations were enough for this simulation.

The gain provided by MAP decoding is clear from fig. 5.14, given that the first iteration corresponds to the receivers that follow the steps of the CF decoder for random codes.

5.7.4 Analysis of the quantizer

In our iterative receiver, the quantizer design affects exchanges of information in two directions : the dequantizer transfers extrinsic information on $\boldsymbol{U_r}$ into a priori information on $\boldsymbol{S_s}$, and conversely transfers extrinsic information on $\boldsymbol{S_s}$ into a priori information on $\boldsymbol{U_r}$. The design of the partition of the quantizer should thus ensure that the quantizer makes the most of these two-way transfers, the final performance measure being the ability to decode $\boldsymbol{U_s}$ successfully.

This section analyzes these transfers in terms of mutual information, a common practice for analyzing the behavior of iterative decoding algorithms (see section 2.5.2). The two extrinsic information transfer functions (EXIT functions) can be used along with the EXIT functions of the constituent codes to analyze the behavior of the decoder and choose system parameters that enable successful decoding.

The EXIT functions for the dequantizer must describe the transfer from constituent decoder to constituent decoder. For our receiver (fig. 5.13), the transfer functions include thus both the dequantizer and the interleaver. Because of the interleaver, the correlation between bits $\boldsymbol{U_r}$ does not have to be considered.

Bit flipping

The computation of the mutual informations involving bits $\boldsymbol{U_r}$ must be handled with care, because these are in general not equiprobable. The method we use for dealing with these biased bits is bit flipping as suggested in [85]. This consists in defining an equivalent system where biased bits $\boldsymbol{U_r}$ are pseudo-randomly flipped into a sequence

of equiprobable bits $\tilde{\boldsymbol{U}}_r$. Given a sequence $\boldsymbol{F}$ of independent and uniformly distributed bits, flipping is as follows :

$$\tilde{\boldsymbol{U}}_r = \boldsymbol{U}_r + \boldsymbol{F}. \tag{5.46}$$

The equivalent system uses this sequence $\tilde{\boldsymbol{U}}_r$ instead of $\boldsymbol{U}_r$ as input to the encoder and modulator. The receiver knows the sequence $\boldsymbol{F}$ and converts the log-ratios on bits $\tilde{\boldsymbol{U}}_r$ into log-ratios on $\boldsymbol{U}_r$ where needed. By linearity of the codes and symmetry of the channel, the flipped system is equivalent to the original one. The advantage of using this trick here is that the relay code needs to be characterized only by its EXIT function for equiprobable bits, whatever the bias in $\boldsymbol{U}_r$. Changes in the bias of $\boldsymbol{U}_r$ are accounted for by the two EXIT functions of the dequantizer instead. This enables fair comparisons of quantizers from the comparison of the EXIT functions they yield.

EXIT functions of the dequantizer

We are now ready to define the two EXIT functions of the dequantizer $T_{\tilde{U}_r S_s}$ and $T_{S_s \tilde{U}_r}$, corresponding to (5.41)–(5.42) and (5.43)–(5.44) respectively, along with the mutual informations under consideration (indices n and q have been omitted) :

$$T_{\tilde{U}_r S_s}(I(\tilde{U}_r; L_{e,\tilde{U}_r})) = I(S_s; L_{a,S_s}) \tag{5.47}$$

$$T_{S_s \tilde{U}_r}(I(S_s; L_{e,S_s}), I(\tilde{U}_r; L_{e,\tilde{U}_r})) = I(\tilde{U}_r; L_{a,\tilde{U}_r}). \tag{5.48}$$

Note that the transfer function $T_{S\tilde{U}_r}$ depends on informations coming from both constituent decoders, as (5.43)–(5.44) show.

Analytical treatment of these two input-output transfer functions is difficult, so we resort to numerical simulations to calculate them. This empirical approach is made possible by the two following observations, made in [35]. First, the distribution of the extrinsic log-ratios $L_{e,\tilde{U}_r}$ or L_{e,S_s} approaches a Gaussian-like distribution with increasing number of iterations; and second, the interleaver ensures that the log-ratios $L_{e,\tilde{U}_r}$ remain fairly uncorrelated, locally, over many iterations. For these reasons, one can generate input sequences of extrinsic log-ratios as if they were made of independent Gaussian-distributed log-ratios. One determines points of the EXIT function by computing the empirical mutual information observed at the output, for several values of the mutual information at the input.

EXIT functions for the quantizer of fig. 5.11 and two different partitions are presented in fig. 5.15, for $\frac{E_s}{N_0} = -1$ dB at the relay.

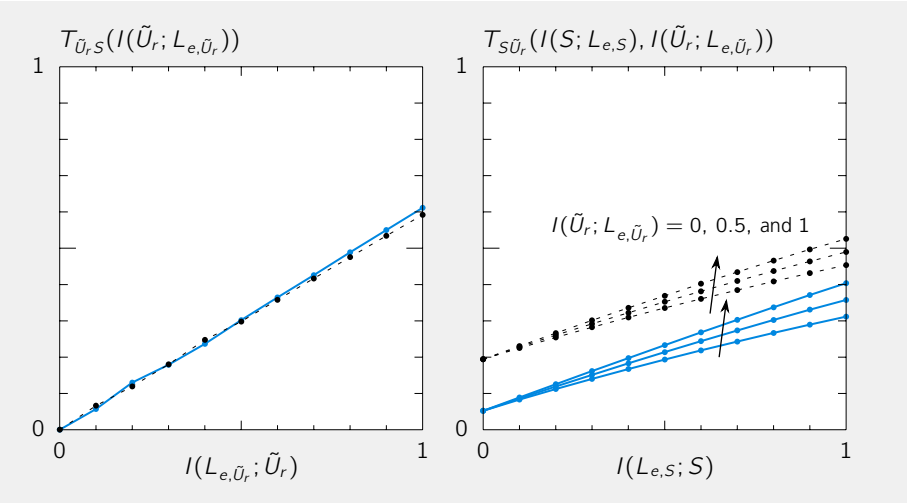


Figure 5.15: EXIT functions $T_{\tilde{U}_r S}(\cdot)$ (left) and $T_{S\tilde{U}_r}(\cdot,\cdot)$ (right, for three values of the second argument $I(\tilde{U}_r; L_{e,\tilde{U}_r})$). The solid blue curves correspond to a quantizer partition that maximizes $I(S; W_r)$ and thus $T_{\tilde{U}_r S}(1)$. The dashed black curves characterize a quantizer that produces a lower value of $I(S; W_r)$ but also a lower entropy $H(W_r)$. The lower entropy causes the vertical shift of $T_{S\tilde{U}_r}(\cdot,\cdot)$.

Particular points of these curves can be related to two quantizer properties, namely the mutual information between its inputs and outputs $I(S_s; W_r)$, and the entropy of its outputs $H(W_r)$. We have

$$T_{\tilde{U}_r S_s}(1) = I(S_s; W_r), \tag{5.49}$$

because $I(\tilde{U}_r; L_{e,\tilde{U}_r}) = 1$ involves that the bits $U_{r,n}^q$ and thus W_r are known with certainty; in that case only one term is non-zero in (5.42) and $P(L_{a,S_s}|S_s)$ is distributed as $P(W_r|S_s)$. In the appendix we also show that, for the quantizer of fig. 5.11, $T_{S_s\tilde{U}_r}(0,0)$ increases when $H(W_r)$ decreases, and conversely (the exact relation depends on the labeling function $\mathcal{L}(\cdot)$). In fig. 5.15, the quantizer corresponding to the solid blue curves maximizes $I(S_s; W_r)$. The quantizer corresponding to the dashed black curves reaches a lower $I(S_s; W_r)$ but has a lower entropy $H(W_r)$, thus more favorable functions $T_{S_s\tilde{U}_r}(\cdot,\cdot)$ (i.e., bits $\boldsymbol{U_r}$ are more biased, so that the a priori information used by the source-channel code is higher).

Comparison with other criteria for quantizer design

In comparison with our approach, the quantizer design criterion of [75] corresponds to choosing the partition that maximizes $I(S_s; W_r)$, the upper right point of the transfer

function $T_{\tilde{U}_r S_s}(\cdot)$.

The design criterion of [69] is the same, but constraints the entropy $H(W_r)$ to be lower than the available rate on the relay-to-destination channel. Such an optimization addresses the same compromise as here between $T_{\tilde{U}_r S_s}(\cdot)$ and $T_{S_s \tilde{U}_r}(\cdot,\cdot)$. They require however that $\boldsymbol{U_r}$ be decoded successfully in just one iteration. The dual criterion is addressed in [74].

Finally, in [76] the quantizer is chosen so as to maximize the achievable rate of the proposed Compress-Forward system. The system is similar to ours, with a scalar quantizer followed by a Slepian-Wolf source-channel code; the decoder however is not iterative. The same rate optimization can be achieved with EXIT charts, using the EXIT functions of the dequantizer along with those of the constituent decoders of $\boldsymbol{U_s}$ and $\boldsymbol{U_r}$. The EXIT charts approach enables to handle the iterative nature of the receiver.

5.7.5 Comparison between quantization of channel observations or soft decoder outputs

Practical Compress-Forward schemes found in the literature have considered either the compression of the signal received by the relay, $\boldsymbol{Y_{sr}}$, or the compression of the outputs $\boldsymbol{L_{U_s}}$ of a soft-input/soft-output decoder at the relay (e.g. [75, 7, 74]). The latter option is motivated by the idea that the soft decoder outputs should contain all relevant information about $\boldsymbol{U_s}$ present in $\boldsymbol{Y_{sr}}$; the soft decoder can be considered as a vector pre-processer before the scalar quantizer. Note that the case studied in section 5.6 corresponds to the particular case of binary quantization of soft decoder outputs.

As no other paper has compared these two options, this section carries out a comparison for the particular coded scheme presented above. The system of section 5.7.3 is compared to a similar system that decodes $\boldsymbol{Y_{sr}}$ into a sequence of soft log-ratios $\boldsymbol{L_{U_s}}$, and quantizes them instead of $\boldsymbol{Y_{sr}}$ (fig. 5.16). The relay is located at a normalized distance $d = 0.9$ on the line joining the user and the destination. Quantizers use partitions with 2, 4, or 16 bins. The relay-to-destination channel is assumed to be perfect, i.e., the quantizer outputs are perfectly known by the destination.

Proper comparison must take into account the fact that the five quantizers of fig. 5.16 produce different entropies of $\boldsymbol{U_r}$, for two reasons. The first reason is that the five quantizers produce different entropies $H(W_r)$, and the second reason is that the sequences $\boldsymbol{Y_{sr}}$ and $\boldsymbol{L_{U_s}}$ have different lengths : quantizing $\boldsymbol{Y_{sr}}$ yields $K \cdot Q$ bits U_r

whereas quantizing $\boldsymbol{L}_{U_s}$ produces $N \cdot Q$ bits U_r. These simulations use a code of rate $\frac{1}{2}$ at the source, thus $N = 2 \cdot K$.

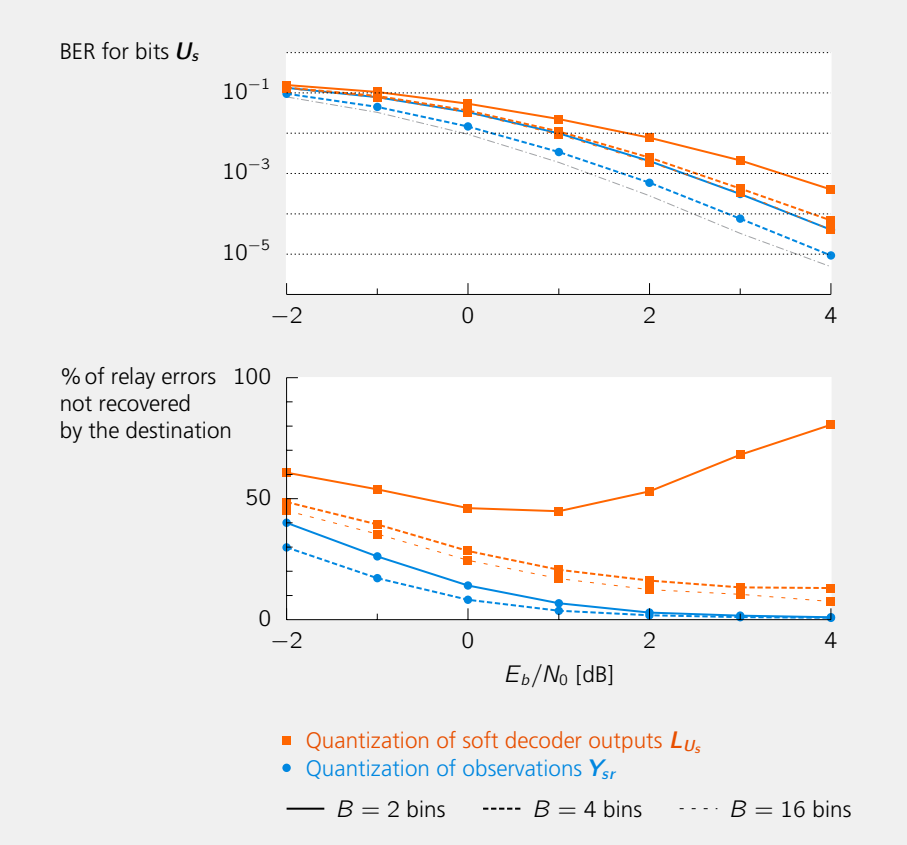


Figure 5.16: Comparison of several coding schemes at the relay, with quantization of soft decoder outputs or channel observations. The relay-to-destination link is considered perfect. Four-levels quantization of observations achieves a bit error rate close to perfect unquantized transmission (gray dashed curve in the upper graph). The lower figure suggests that errors made by a soft decoder tend to propagate.

The upper part of fig. 5.16 shows that quantizing the observations $\boldsymbol{Y}_{sr}$ is always better than quantizing the soft decoder outputs $\boldsymbol{L}_{U_s}$, when the relay-to-destination transmission is perfect. Quantizing $\boldsymbol{L}_{U_s}$ might still have some interest in our case if the relay-to-destination link is weak, because its can produce lower entropies for $\boldsymbol{U}_r$ (essentially because our codes choices make the sequence $\boldsymbol{L}_{U_s}$ shorter).

The lower part of fig. 5.16 carries the comparison further by looking at the positions of errors, at the relay and at the destination. Errors at the relay are defined as sign

errors of the soft outputs $\boldsymbol{L}_{U_s}$, or equivalently as binary errors in hard decisions based on $\boldsymbol{L}_{U_s}$. To provide a point of comparison, we use this same definition of relay errors for the relay quantizing $\boldsymbol{Y}_{sr}$, even though it is not supposed to calculate $\boldsymbol{L}_{U_s}$. Fig. 5.16 suggests that, in the case of soft decoder output quantization, errors made at the relay are difficult to recover at the destination. In other words, errors made when attempting to decode at the relay are the most favorable places for errors at the destination, i.e., errors propagate.

Part of the reason for explaining the poorer performance of quantizing $\boldsymbol{L}_{U_s}$ instead of $\boldsymbol{Y}_{sr}$ is that the equivalent channel between $\boldsymbol{S}_s$ and $\boldsymbol{L}_{U_s}$ is not memoryless, whereas the receivers presented so far assume it is.

In conclusion, this shows that, at least for the system considered here, preceding the quantizer with a soft decoder can be more harmful than beneficial.

5.8 PERFORMANCE COMPARISONS FOR HYBRID SCHEMES

The preceding sections have presented several relaying techniques and the corresponding receivers; this section compares them.

Comparisons are carried out for transmission on wireless channels, without channel state information at the transmitters. We also choose to use hybrid transmission schemes, where the relay can switch between two protocols depending on the quality of the signal it receives :

- If the relay successfully decodes the error-correcting code, it uses a regenerative protocol (i.e., DF, section 5.3).
- If the relay is not able to decode successfully, it uses one of the non-regenerative protocols : AF (section 5.4), Soft-DF (section 5.5), DF with the error-resilient reception (section 5.6), or CF with iterative MAP reception (section 5.7).

Comparisons depend on the geometric configuration of the relay system. We provide results for two distinct situations, either when the relay is at a distance $d = 0.6$ from the source terminal on the line joining the source terminal and the destination, and when the relay is at a distance $d = 0.9$ from the source terminal.

Our simulations suggest that the hybrid scheme using CF as non-regenerative protocol tends to work best. More qualified results are presented in the following sections.

5.8.1 Simulation setup

Wireless channels are modeled as Rayleigh block-fading channels. All three terminals are lined up, the relay being located between the source terminal and the destination. Path losses between the terminals determine the power levels at the receivers; the path loss exponent is chosen equal to 3.5. The two-sided power spectral densities of noise $N_0/2$ are set equal for both the relay and destination receivers.

In all schemes, the source terminal encodes its data blocks of length $K = 512$ bits with an RSC encoder of rate $1/2$, constraint length 5 and generator polynomials $(23_8, 37_8)$ (octal notation). The source terminal uses BPSK modulation. It transmits thus $N = 2 \cdot K$ symbols, and the relay terminal forwards K additional symbols.

In all hybrid schemes, when using DF, the relay uses a random interleaver and re-encodes the data with the same RSC encoder as the source terminal. As for the parameters of the non-regenerative protocols, they are given separately for each case addressed below.

The simulations use as reference system an hybrid transmission scheme using, as non-regenerative protocol, the CF protocol with binary quantization at the relay (as described in section 5.7.3). The outputs of the quantizer are encoded with BICM : they are encoded by the same recursive encoder as the source terminal; then the resulting parity bits are randomly interleaved and forwarded using 4-ASK modulation with set-partitioning mapping (systematic bits are not forwarded).

For protocols using iterative receivers, the results correspond to the fifth iteration. Moreover, for the iterative reception of CF, the BICM decoder makes two iterations within each global iteration.

5.8.2 Performance comparisons for the non-regenerative part of the protocol

Figures 5.17–5.20 present four comparisons of hybrid schemes. Each of this figures contains, in the left column, the results for a relay at distance $d = 0.6$ from the source terminal, and in the right column, the results for a relay at distance $d = 0.9$.

In each column, the first figure is a comparison in terms of bit error rate, and the second is a comparison in terms of frame error rate. *These error rates are not for the whole hybrid scheme, they consider only frames sent with the non-regenerative protocol (frames that the relay could not decode successfully).* Frames relayed with

DF are thus not taken into account. The lowest figure shows the usage of the non-regenerative protocol, which corresponds to the percentage of unsuccessful frame decoding at the relay.

BER and FER curves are plotted as a function of E_b/N_0, where E_b is the total energy per bit spent by the source and the relay all together.

CF with binary quantization vs. AF

As shown in fig. 5.17, CF is better than AF both in terms of bit error rate (gain larger than 2 dB) and in terms of frame error rate, for both locations of the relay. This contrasts with the information theoretic results of section 4.6.1, where AF was shown to be better. The conclusions made for random Gaussian codes can thus not be extended to practical systems using binary codes.

Note that the results of section 4.6.1 are obtained under slightly different assumptions. First, they assume that the relay forwards as many symbols as the source terminal, whereas here the relay forwards half the number of symbols sent by the source. We have verified however that the results of section 4.6.1 hold when using the latter repartition of channel uses. Next, the two following differences make the CF protocol used here less sophisticated than the one of section 4.6.1 :

- The practical scheme of this section uses simple *scalar* quantization followed by a binary source-channel code. In contrast, the information theoretical scheme used in section 4.6.1 uses (vector) Wyner-Ziv coding with a random Gaussian codebook (which is not optimal but is the best known compression method — or rather, the only one investigated in this context— when using a Gaussian codebook at the source).
- The system of this section is not optimized, whereas the system of section 4.6.1 optimizes the parameters of CF forwarding at each frame (code rate R_r and compression noise N_q).

Thus, the CF protocol of this section manages to outperform AF even though it is rather simple. And finally, as a last difference, the CF system of this section uses the iterative reception algorithm of section 5.7, which, as noted in section 5.7.2, bears some structural differences with respect to the information-theoretic decoding algorithm. In particular, it uses the structure of the error-correcting code to refine the side

information used for decompression.

CF with binary quantization vs. Soft-DF

The comparisons in fig. 5.18 show that Soft-DF performs almost always worse than CF with iterative reception.

Comparing the performance of Soft-DF with that of AF (compare with fig. 5.17) gives some insight on the advantages and disadvantages of Soft-DF. Soft-DF is better in terms of BER, but worse in terms of FER. Using a soft decoder at the relay helps improving the information forwarded on the bits, since the BER is lower. However, while producing less bit errors, Soft-DF produces more frame errors. It seems that the decoder at the destination hardly recovers errors produced by the decoder at the relay : the frame error rate at the destination is pulled down by the frame error rate at the relay.

CF with binary quantization vs. DF with error-resilient receiver

As shown in fig. 5.19, DF can achieve better or almost equal BER as CF when using the receiver resilient to relay errors presented in section 5.6 (even though the error rates of fig. 5.19 are for frames not successfully decoded by the relay). However, CF works much better in terms of frame error rate. Again, trying to decode at the relay improves the bit error rate at the destination but pulls down the frame error rate, because the errors made by the decoder at the relay are hardly recovered by the decoder at the destination.

Using DF with the receiver resilient to relay errors is especially useful when the relay is farther away from the destination (first column in fig. 5.19), because the number of bits to transmit by the relay is lower than with CF (see discussion in section 5.7.5). In this simulation, the CF and DF protocols forward the same number of symbols, but CF uses 4-ASK while DF uses BPSK.

CF with binary quantization vs. CF with 4-levels quantization

Figure 5.20 compares the CF protocol with binary quantization with the following CF protocol with 4-levels quantization. The partition of the scalar quantizer is chosen so as to maximize the mutual information $I(S_s; W_r)$ (see section 5.7.4). The binary labeling of the bins is the same as in fig. 5.11, and the relay uses 16-ASK modulation with set-partitioning mapping.

As intuition suggests, CF with 4-levels quantization works better when the relay is closer to the destination, because the relay-to-destination link is better in that case and can support a higher transmission rate.

Conclusions

We conclude this section with some additional comments drawn from fig. 5.17–5.20.

First, in no situation is the AF protocol the best scheme. This contrasts with the information-theoretical results of section 4.6.1.

Next, Soft-DF is always better than AF in terms of BER, and very close to AF in terms of FER. It is however outperformed by DF with error-resilient receiver, and most often by CF.

Finally, CF with 2 or 4-levels quantization performs better than any other protocol in terms of FER. It is also always better in terms of BER, behalves when compared to DF with error-resilient receiver, when the relay-to-destination is weak.

5.8.3 Performance comparisons for the whole protocol

To complete these simulation results, this section presents the overall performance of hybrid protocols, i.e., the performance when both frames sent with DF and frames sent with the non-regenerative protocol are taken into account.

Figure 5.21 compares the hybrid protocols that have respectively AF and CF as non-regenerative protocol. The DF-AF protocol is taken as a reference since it corresponds to a usual choice of hybrid protocol, and the DF-CF protocol is chosen because CF was identified as the most promising non-regenerative protocol in the previous section, due to its good performance both in BER and in FER. In the DF-CF protocol, the CF part of the protocol uses two-levels quantization when the relay is located at $d = 0.6$, and four-levels quantization when the relay is located at $d = 0.9$. The filled area delimits the error rates achievable by allowing cooperation : the upper bound corresponds to non-cooperative transmission and the lower bound corresponds to transmission with perfect knowledge of $\boldsymbol{Y}_{sr}$ at the destination (equivalent to single-antenna to two-antennas transmission).

The DF-CF protocol outperforms the DF-AF protocol in terms of bit error rate and of frame error rate, for both locations of the relay. Moreover, the error rates of the DF-CF

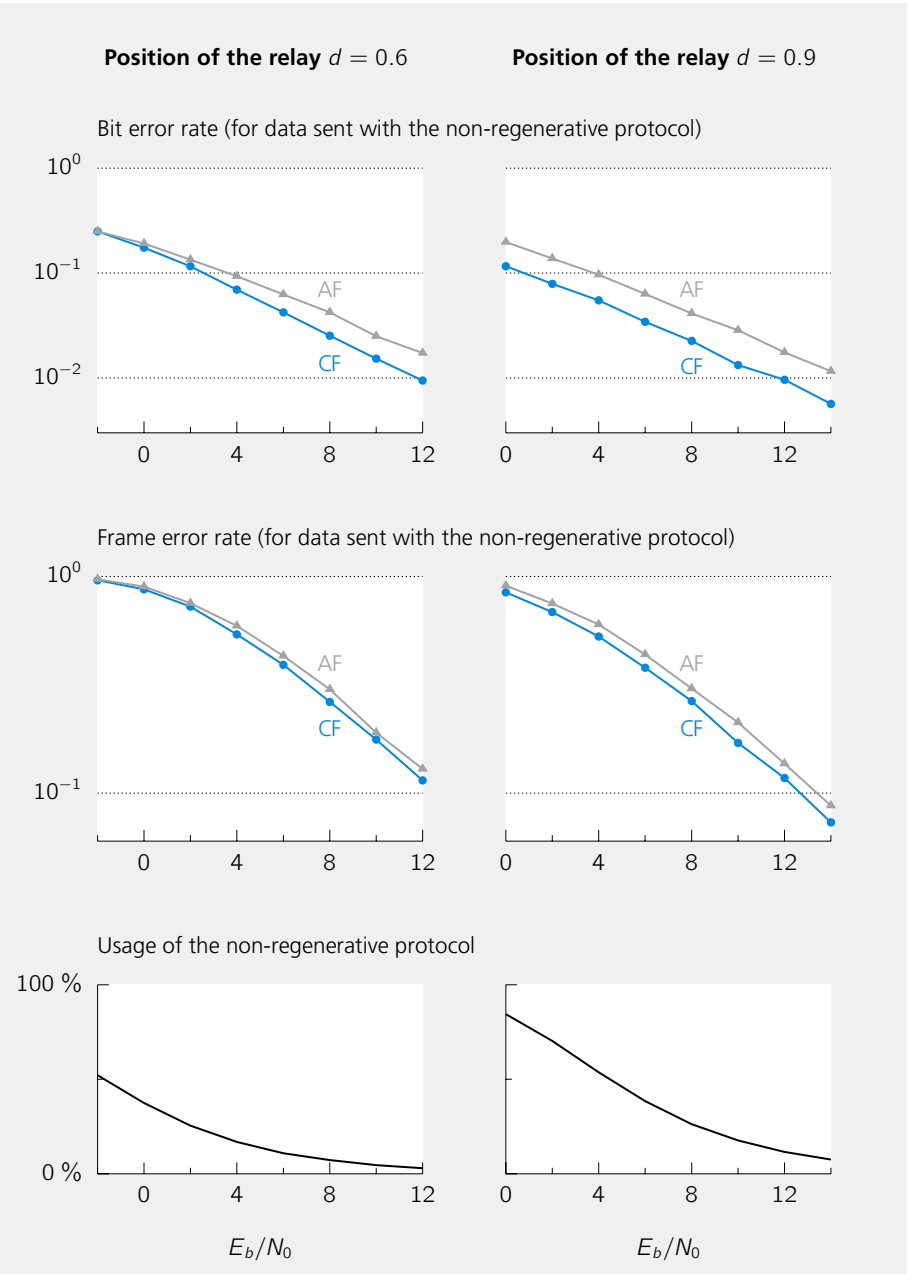


Figure 5.17: Comparison of two hybrid schemes using, as non-regenerative protocol, either AF or CF with binary quantization. The error rates are for frames sent with the non-regenerative protocol only.

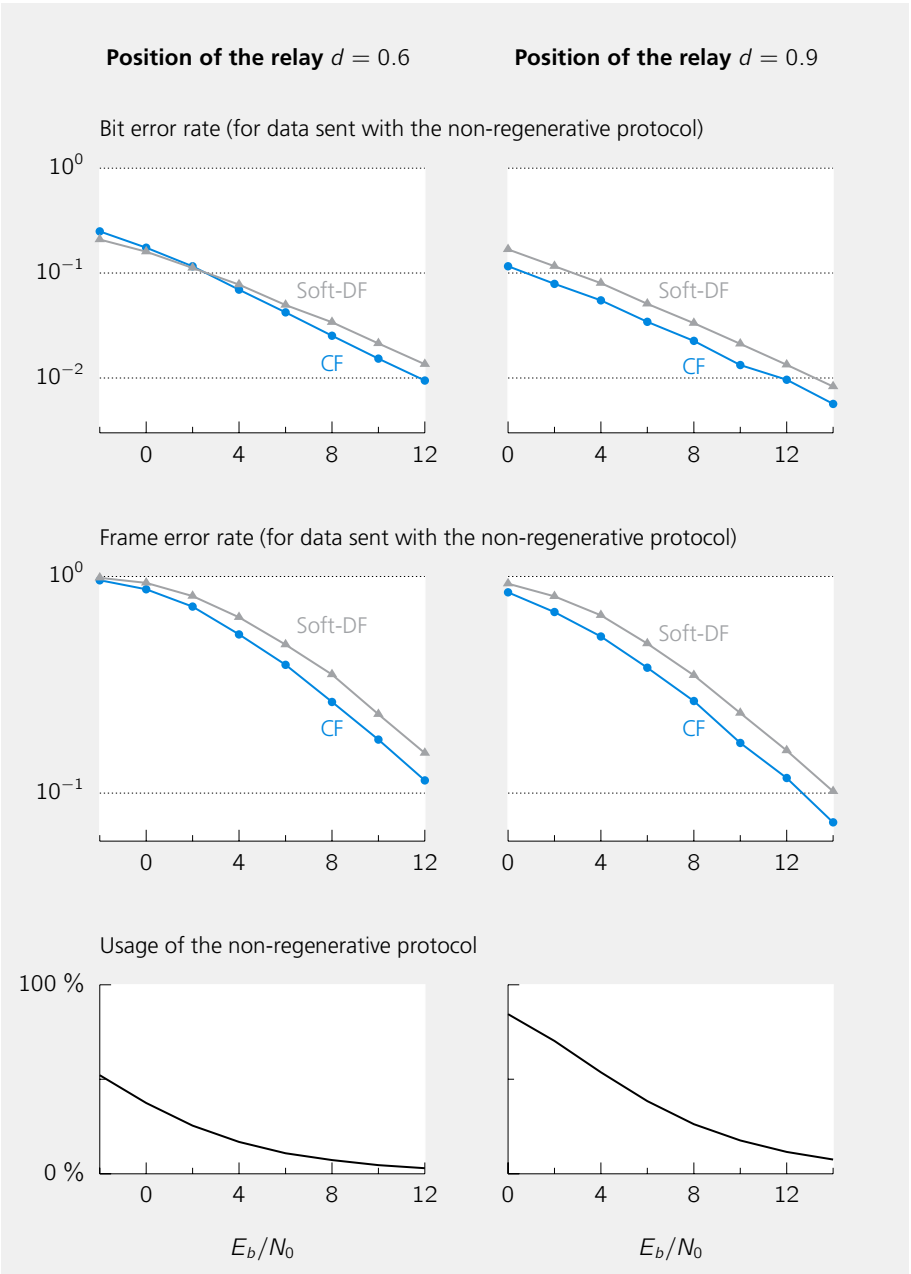


Figure 5.18: Comparison of two hybrid schemes using, as non-regenerative protocol, either Soft-DF or CF with binary quantization. The error rates are for frames sent with the non-regenerative protocol only.

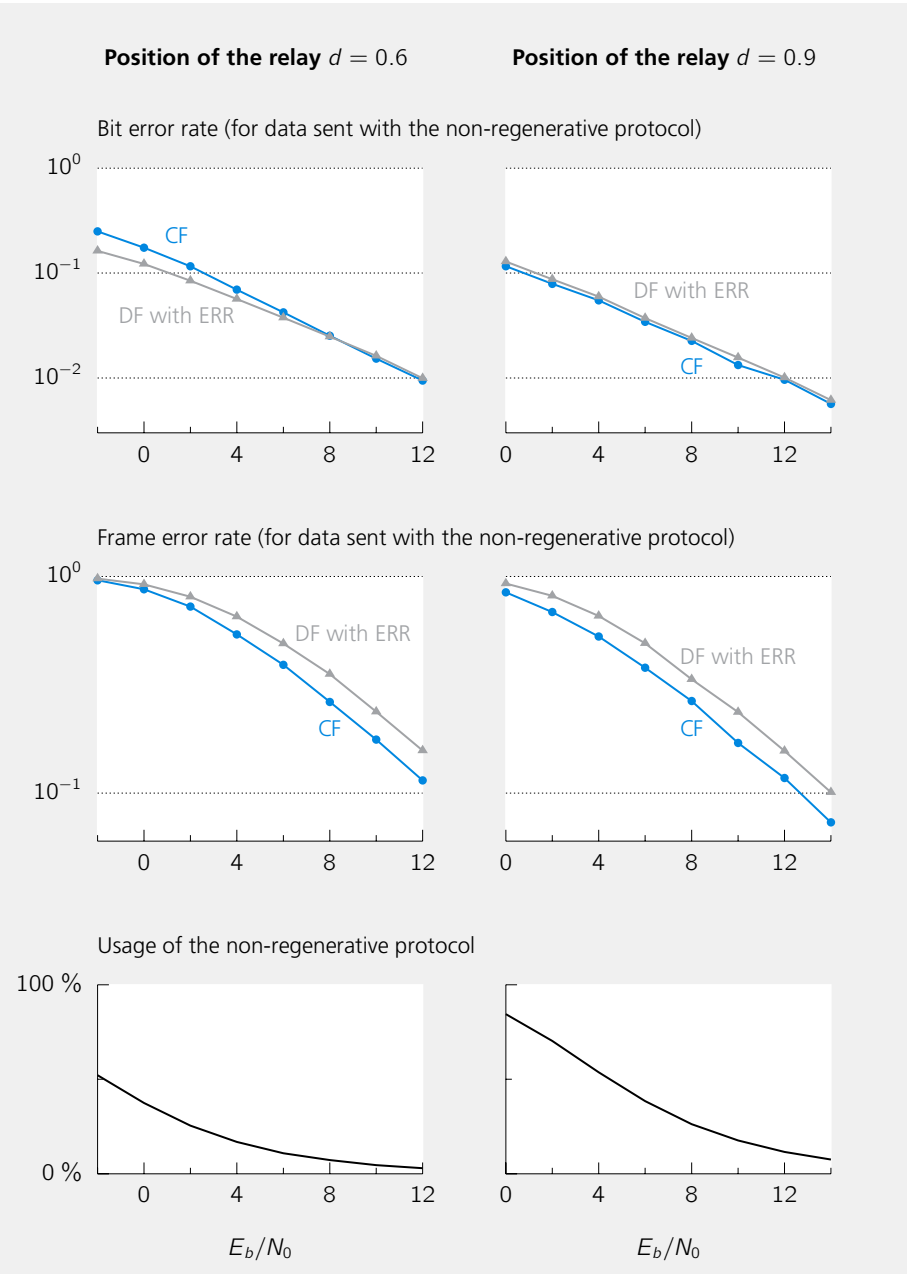


Figure 5.19: Comparison of two hybrid schemes using, as non-regenerative protocol, either DF with an error-resilient receiver or CF with binary quantization. The error rates are for frames sent with the non-regenerative protocol only.

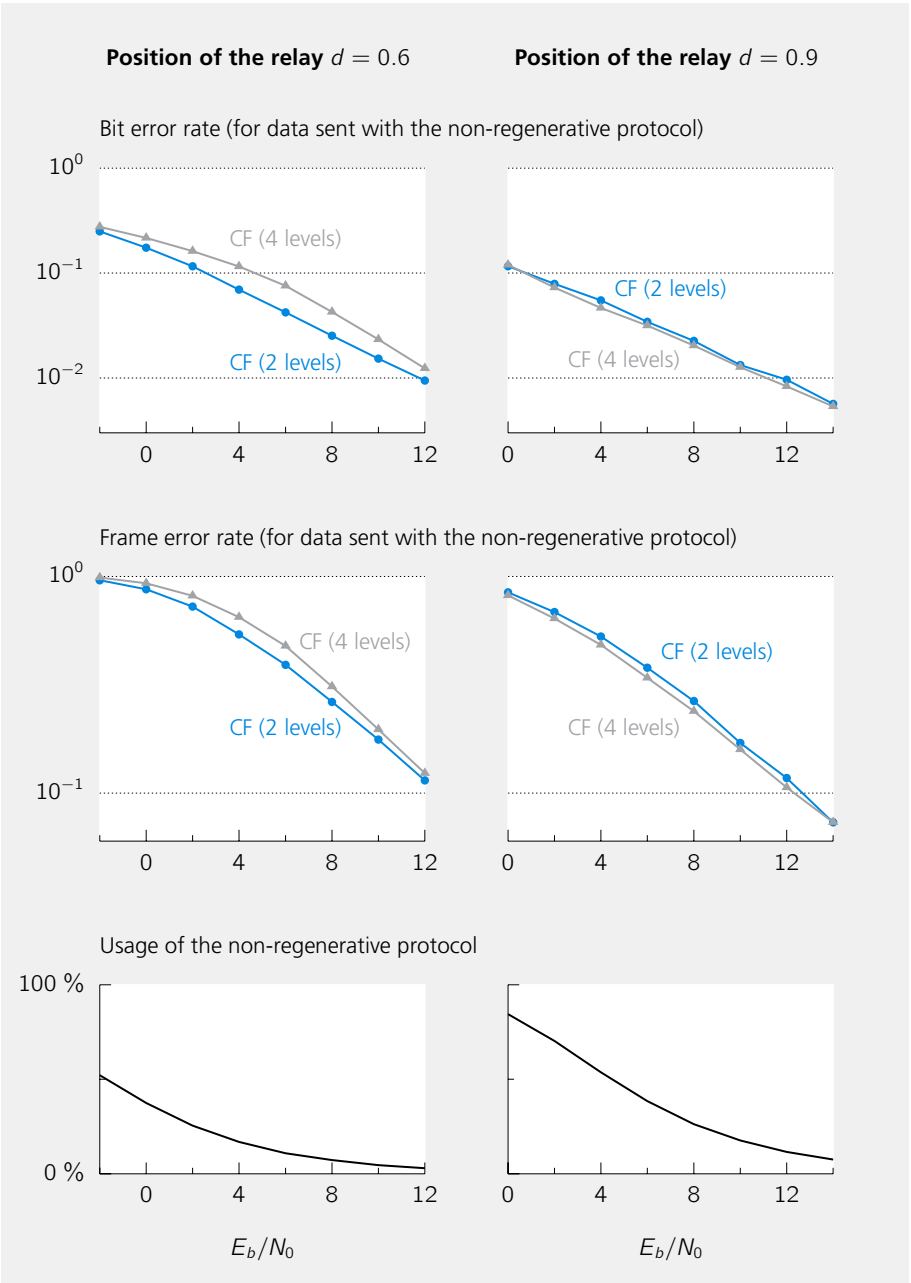


Figure 5.20: Comparison of two hybrid schemes using, as non-regenerative protocol, either CF with binary quantization or CF with 4-levels quantization. The error rates are for frames sent with the non-regenerative protocol only.

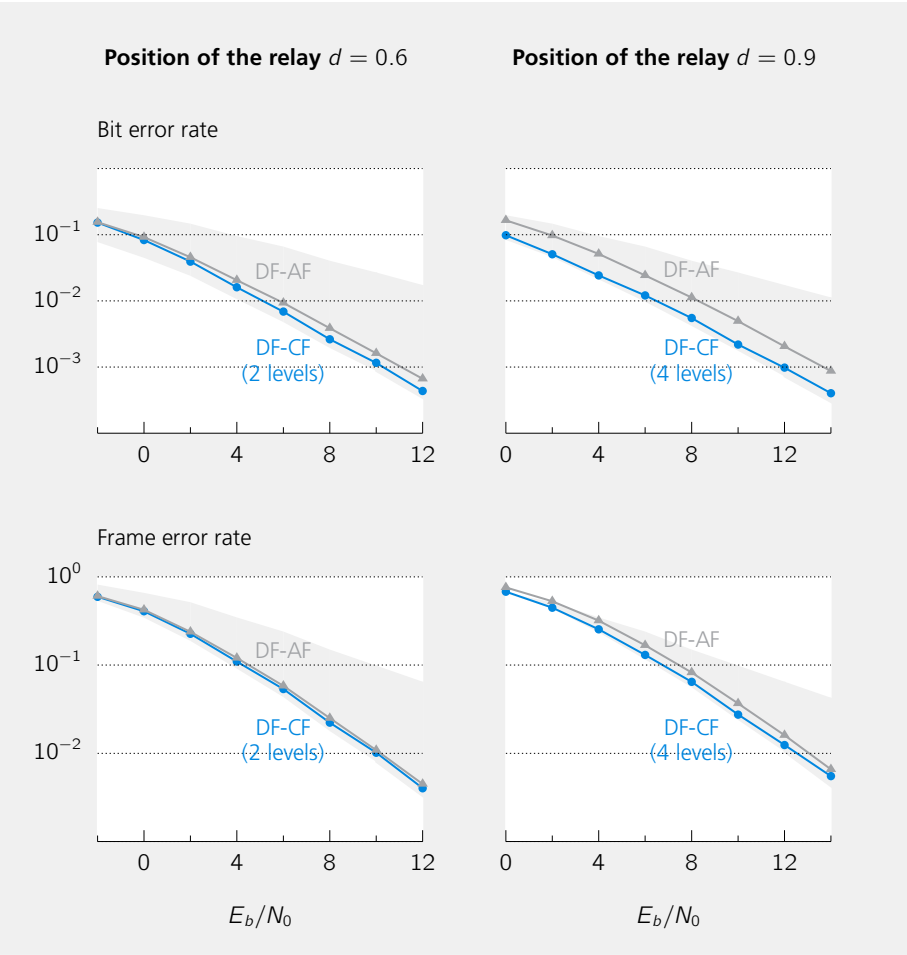


Figure 5.21: Comparison of two hybrid schemes using, as non-regenerative protocol, either AF or CF. The hybrid DF-CF protocol achieves lower bit error rates and frame error rates. Moreover, the error rates achieved by this protocol are very close to the lower bound (lower limit of the filled region).

protocol are very close to the lower bound corresponding to perfect knowledge of $\boldsymbol{Y}_{sr}$ at the destination.

An important conclusion of all these simulations, already observed and discussed in section 5.8.2 (comparison between AF and CF), is that an hybrid DF-CF protocol can perform better than an hybrid DF-AF protocol, even though information-theoretical results would predict the opposite (see section 4.6.1). Reasons for the difference between information-theoretic results and these practical schemes is most probably related to a difference in channel input alphabets : the information-theoretic results use Gaussian codes, whereas these practical transmission schemes use finite symbol alphabets (here, BPSK or M-ASK). This emphasizes that comparisons between these two approaches should be carried out with care.

Appendix 5.A LINK BETWEEN THE ENTROPY OF QUANTIZER OUTPUTS AND THE DEQUANTIZER TRANSFER FUNCTIONS

This appendix shows how the entropy $H(W_r)$ is related to the value of the EXIT transfer function $T_{S\tilde{U}_r}(0,0)$, for the quantizer of fig. 5.11. The relation is not general because it depends on the labeling $\mathcal{L}(\cdot)$, but it is provided here to give the intuition behind the EXIT functions of fig. 5.15.

We first calculate the entropy of the quantizer indices W_r. The distribution of W_r can be parametrized as follows, given the symmetry of the signal to quantize and the symmetry of the quantizer :

$$P_{W_r}(1) = P_{W_r}(4) = p \qquad 0 \leq p \leq \frac{1}{2} \tag{5.50}$$

$$P_{W_r}(2) = P_{W_r}(3) = \frac{1}{2} - p. \tag{5.51}$$

Consequently we have

$$H(W_r) = 2\left[-p\log_2 p - (\frac{1}{2} - p)\log_2(\frac{1}{2} - p)\right] = 1 + H_b(2p) \tag{5.52}$$

where $H_b(x) \triangleq -x\log_2 x - (1-x)\log_2(1-x)$.

The index W_r is translated into a binary label $U_r^1 U_r^2$. Given the labeling defined in fig.

5.11, these are distributed as follows :

$$P_{U_r^1}(1) = 2p \tag{5.53}$$

$$P_{U_r^2|U_r^1}(1|1) = P_{U_r^2|U_r^1}(1|-1) = \frac{1}{2}, \tag{5.54}$$

which makes clear that bits U_r^1 are in general biased.

We now turn to the dequantizer, to evaluate the transfer function (5.48) :

$$T_{S\tilde{U}_r}(I(S;L_{e,S}), I(\tilde{U}_r;L_{e,\tilde{U}_r})) = I(\tilde{U}_r;L_{a,\tilde{U}_r}),$$

when $I(S;L_{e,S}) = I(\tilde{U}_r;L_{e,\tilde{U}_r}) = 0$. Taking thus $L_{e,S} = L_{e,\tilde{U}_r} = 0$, the output log-ratios produced by (5.43)–(5.44) are equal to :

$$L_{a,U_r^1} = \log\frac{2p}{1-2p} \triangleq I \tag{5.55}$$

$$L_{a,U_r^2} = 0. \tag{5.56}$$

Bits $U_r^1 U_r^2$ are then randomly interleaved; at the output of the interleaver a bit U_r may correspond either U_r^1 or U_r^2 with equal probability. As a consequence,

$$P_{U_r}(1) = \frac{P_{U_r^1}(1) + P_{U_r^2}(1)}{2} = p + \frac{1}{4} \tag{5.57}$$

and

$$H(U_r) = H_b(p + \frac{1}{4}). \tag{5.58}$$

Similarly, a log-ratio L_{a,U_r} at the output of the interleaver may correspond either to L_{a,U_r^1} or L_{a,U_r^2} with equal probability; thus

$$P_{L_{a,U_r}}(0) = P_{L_{a,U_r}}(I) = \frac{1}{2}. \tag{5.59}$$

The bits U_r are then randomly flipped, and the log-ratios are correspondingly changed sign. After flipping, log-ratios that where equal to zero do not change, thus

$$P_{L_{a,\tilde{U}_r}|\tilde{U}_r}(0|\tilde{U}_r) = \frac{1}{2} \qquad \text{for } \tilde{U}_r = -1, 1. \tag{5.60}$$

Half of the log-ratios that were equal to I are not flipped ($\tilde{U}_r = U_r$) :

$$P_{L_{a,\tilde{U}_r}|\tilde{U}_r}(I|1) = \frac{P_{U_r}(1)}{2}, \quad P_{L_{a,\tilde{U}_r}|\tilde{U}_r}(I|-1) = \frac{P_{U_r}(-1)}{2}, \tag{5.61}$$

and half of them are flipped ($\tilde{U}_r \neq U_r$) :

$$P_{L_{a,\tilde{U}_r}|\tilde{U}_r}(-I|1) = \frac{P_{U_r}(-1)}{2}, \quad P_{L_{a,\tilde{U}_r}|\tilde{U}_r}(-I|-1) = \frac{P_{U_r}(1)}{2}. \tag{5.62}$$

We also have

$$P_{L_{a,\tilde{U}_r}}(0) = \frac{1}{2} \qquad \text{and} \tag{5.63}$$

$$P_{L_{a,\tilde{U}_r}}(I) = P_{L_{a,\tilde{U}_r}}(-I) = \frac{1}{4}. \tag{5.64}$$

Finally, the mutual information $I(\tilde{U}_r; L_{a,\tilde{U}_r})$ is equal to :

$$\begin{aligned} I(\tilde{U}_r; L_{a,\tilde{U}_r}) &= H(L_{a,\tilde{U}_r}) - H(L_{a,\tilde{U}_r}|\tilde{U}_r) \\ &= \frac{3}{2} - 1 - \frac{1}{2}H_b(P_{U_r}(1)) \\ &= \frac{1}{2} - \frac{1}{2}H_b(p + \frac{1}{4}). \end{aligned} \tag{5.65}$$

Comparing (5.52) and (5.65) shows that $T_{S\tilde{U}_r}(0,0)$ increases when $H(W_r)$ decreases and vice versa.

Conclusions

The last chapter of this thesis has presented three techniques for transmission over the orthogonal relay channel. All three techniques aim at exploiting data frames that the relay fails to decode.

First, the Soft-DF cooperation technique (section 5.5) deals with uncertainty at the relay by avoiding taking hard decisions on the data. It performs instead the operations of a DF relay in a soft fashion; this was made possible by the design of a SISO encoder. As a consequence, the Soft-DF technique enables to regenerate the signal as a DF relay does, while keeping the reliability information of the decoded bits, as an AF relay does. It achieves better performance than these two protocols, as simulations show.

Second, the error-resilient decoding algorithm for DF (section 5.6) enables the use of turbo-coded DF relaying even when the relay forwards sequences containing errors. This strategy provides a sensible gain of performance over usual AF and DF strategies, as shown in simulations. Of course, the probability of error on the source-to-relay link still limits the performance at the destination; this was made clear by analyzing the information exchanges between constituent decoders during the iterative decoding process.

And third, the MAP CF decoding algorithm (section 5.7) improves the performance of practical communication with the CF protocol, and shows that the structure of the CF decoding algorithm for random codes is not well suited to the practical CF implementations found in the literature. In particular, the proposed algorithm enables to refine the side information used for Wyner-Ziv decoding by using the error correcting code of the source signal, whereas usual algorithms do not. In addition to improving the performance of CF decoding, the proposed algorithm also yields a criterion for quantizer design : in a scenario where the relay has full channel state information, the EXIT functions of the dequantizer could be used to choose code and quantizer

parameters that match each other.

Comparisons of these three techniques and AF in the context of hybrid schemes have shown that the CF protocol with the MAP decoding algorithm does most often achieve the best performance. This conclusion is opposed to the theoretical results of chapter 4 where, under similar conditions, the AF protocol was shown to achieve lower outage than the CF protocol (as long as the relay had receiver CSI only). Thus, the results obtained for Gaussian random codes do not necessarily extend to discrete modulation alphabets.

Considering side by side the information-theoretical results for the CF protocol and the MAP decoding algorithm for CF offers some interesting perspectives. As stated above, the MAP decoding algorithm uses the error-correcting code of the source-to-destination signal to enhance the side-information used for Wyner-Ziv decoding. In contrast, the CF decoding algorithm for Gaussian random code does not use such joint decoding; whether this feature can be incorporated in the information-theoretic coding scheme is probably worth investigation.

As to the design of practical coding schemes, systems using the CF protocol have received comparatively less attention than those using the DF protocol. Whereas this thesis has focused on low-complexity systems, perspectives include the design of more complex schemes with stronger codes. Moreover, the MAP CF algorithm derived here, with its new structure, has offered a new way to achieve Wyner-Ziv coding that future code designs might take advantage of.

Bibliography

[1] H. H. Sneessens, X. Jaspar, C. Herzet, and L. Vandendorpe, "Calculating turbo-decoding performance characteristics for adaptive channel coding," in *Proc. IEEE SPAWC*, Helsinki (Finland), Jun. 2007.

[2] ——, "Predictions on turbo-decoding performance for adaptive channel coding," in *41st Asilomar Conference on Signals, Systems and Computers*, Asilomar (USA), Nov. 2007.

[3] H. H. Sneessens, L. Vandendorpe, and J. N. Laneman, "Adaptive Compress-and-Forward relaying in fading environments with or without Wyner-Ziv coding," in *Proc. IEEE ICC*, Dresden (Germany), Jun. 2009.

[4] ——, "Adaptive Compress-and-Forward relaying in fading environments with or without Wyner-Ziv coding," 2009, in submission.

[5] H. H. Sneessens and L. Vandendorpe, "Soft decode and forward improves cooperative communications," in *Proc. IEEE 3G and Beyond*, London (United Kingdom), Nov. 2005.

[6] ——, "Soft decode and forward improves cooperative communications," in *Proc. IEEE CAMSAP*, Puerto Vallarta (Mexico), Dec. 2005.

[7] H. H. Sneessens, J. Louveaux, and L. Vandendorpe, "Turbo-coded decode-and-forward strategy resilient to relay errors," in *Proc. IEEE ICASSP*, Las Vegas (USA), Apr. 2008.

[8] H. H. Sneessens, L. Vandendorpe, G. N. Karystinos, and A. P. Liavas, "On co-operative protocols with binary input," in *Proc. ICT-MobileSummit*, Stockholm (Sweden), Jun. 2008.

[9] H. H. Sneessens, L. Vandendorpe, and J. N. Laneman, "Iterative receiver for practical Compress-Forward with scalar quantization," 2009, in submission.

[10] F. R. Kschischang, B. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Info. Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[11] F. R. Kschischang and B. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 1, pp. 1–11, 1998.

[12] N. Wiberg, H.-A. Loeliger, and R. Kötter, "Codes and iterative decoding on general graphs," *Europ. Trans. Telecomm.*, vol. 6, pp. 513–525, Sep./Oct. 1995.

[13] N. Wiberg, "Codes and decoding on general graphs," 1996.

[14] B. J. Frey and D. J. C. MacKay, "A revolution : Belief propagation in graphs with cycles," *Advances in Neural Information Processing Systems 10*, dec 1997.

[15] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. of IEEE ICC*, Geneva, May 1993, pp. 1064–1070.

[16] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: turbo-codes," *Communications, IEEE Transactions on*, vol. 44, no. 10, pp. 1261–1271, 1996.

[17] R. G. Gallager, *Low-Density Parity-Check Codes*. M.I.T. Press, 1963.

[18] Y. Weiss, "Correctness of local probability propagation in graphical models with loops," *Neural Computation*, vol. 12, no. 1, pp. 1–41, 2000.

[19] Y. Weiss and W. T. Freeman, "Correctness of belief propagation in Gaussian graphical models of arbitrary topology," *Neural Computation*, vol. 13, no. 10, pp. 2173–2200, 2001.

[20] R. J. McEliece and D. J. C. MacKay, "Turbo-decoding as an instance of Pearl's 'belief propagation' algorithm," *IEEE Journal on selected Areas in Communications*, vol. 16, no. 2, pp. 140–152, Feb. 1998.

[21] G. D. Forney, "Codes on graphs : Recent progress," *Physica A*, vol. 302, pp. 1–13, Dec. 2001.

[22] A. P. Worthen, "Codes and iterative receivers for wireless communication systems," CSP Lab., EECS dpt., University of Michigan, Technical Report, May 2001.

[23] J. G. Proakis, *Digital Communications*, 3rd ed. McGraw-Hill, 1995.

[24] L. 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, no. 2, pp. 284–287, 1974.

[25] G. Ungerboeck and I. Csajka, "On improving data-link performance by increasing the channel alphabet and introducing sequence coding," in *1976 Int. Symp. Inform. Theory*, Ronneby, Sweden, jun 1976.

[26] G. Ungerboeck, "Channel coding with multilevel/phase signals," *Information Theory, IEEE Transactions on*, vol. 28, no. 1, pp. 55–67, Jan 1982.

[27] H. Imai and S. Hirakawa, "A new multilevel coding method using error-correcting codes," *Information Theory, IEEE Transactions on*, vol. 23, no. 3, pp. 371–377, May 1977.

[28] E. Zehavi, "8-PSK trellis codes for a rayleigh channel," *Communications, IEEE Transactions on*, vol. 40, no. 5, pp. 873–884, May 1992.

[29] G. Caire, G. Taricco, and E. Biglieri, "Bit-interleaved coded modulation," *Information Theory, IEEE Transactions on*, vol. 44, no. 3, pp. 927–946, May 1998.

[30] X. Li and J. Ritcey, "Bit-interleaved coded modulation with iterative decoding," *Communications Letters, IEEE*, vol. 1, no. 6, pp. 169–171, Nov 1997.

[31] ——, "Trellis-coded modulation with bit interleaving and iterative decoding," *Selected Areas in Communications, IEEE Journal on*, vol. 17, no. 4, pp. 715–724, Apr 1999.

[32] S. ten Brink, J. Speidel, and R.-H. Yan, "Iterative demapping and decoding for multilevel modulation," in *Global Telecommunications Conference, 1998 (GLOBECOM 98)*, vol. 1, 1998, pp. 579–584 vol.1.

[33] A. Gorokhov and M. van Dijk, "Optimised labeling maps for bit-interleaved transmission with turbo demodulation," in *Proc. Vehicular Technology Conference (VTC Spring)*, vol. 2, 2001, pp. 1459–1463 vol.2.

[34] S. ten Brink, "Iterative decoding trajectories of parallel concatenated codes," in *Proc. 3rd IEEE/ITG Conf. Source Channel Coding*, Munich, Germany, Jan. 2000, pp. 75–80.

[35] ——, "Convergence behavior of iteratively decoded parallel concatenated codes," *Communications, IEEE Transactions on*, vol. 49, no. 10, pp. 1727–1737, 2001.

[36] H. El Gamal and A. R. Hammons, Jr, "Analysing the turbo decoder using the gaussian approximation," *IEEE Trans. Info. Theory*, vol. 47, no. 2, pp. 671–686, 2001.

[37] E. C. V. D. Meulen, "Three-terminal communication channels," *Advances in Applied Probability*, vol. 3, no. 1, pp. 120–154, 1971. [Online]. Available: http://www.jstor.org/stable/1426331

[38] T. Cover and A. El Gamal, "Capacity theorems for the relay channel," *IEEE Transactions on Information Theory*, vol. 25, no. 5, pp. 572–584, 1979.

[39] J. N. Laneman and G. W. Wornell, "Energy-efficient antenna sharing and relaying for wireless networks," in *Proc. IEEE WCNC*, Chicago, IL, 2000, pp. 7 – 12.

[40] J. N. Laneman, G. W. Wornell, and D. N. C. Tse, "An efficient protocol for realizing cooperative diversity in wireless networks," in *Proc. IEEE ISIT*, Washington, DC, 2001, p. 294.

[41] J. N. Laneman, D. N. C. Tse, and G. W. Wornell, "Cooperative diversity in wireless networks: Efficient protocols and outage behavior," *IEEE Transactions on Information Theory*, vol. 50, no. 12, pp. 3062–3080, Dec. 2004.

[42] A. Sendonaris, E. Erkip, and B. Aazhang, "User Cooperation Diversity–Part I: System Description," *IEEE Transactions on Communications*, vol. 51, no. 11, pp. 1927–1938, Nov. 2003.

[43] ——, "User Cooperation Diversity–Part II: Implementation Aspects and Performance Analysis," *IEEE Transactions on Communications*, vol. 51, no. 11, pp. 1939–1948, Nov. 2003.

[44] ——, "Increasing uplink capacity via user cooperation diversity," in *Proc. IEEE International Symposium on Information Theory*, Aug. 1998, p. 156.

[45] D. N. C. Tse, P. Viswanath, and L. Zheng, "Diversity-multiplexing tradeoff in multiple-access channels," *IEEE Trans. Inform. Theory*, vol. 50, pp. 1859–1874, 2004.

[46] M. Yuksel and E. Erkip, "Diversity-multiplexing tradeoff in cooperative wireless systems," in *40th Annual Conference on Information Sciences and Systems*, Mar. 2006.

[47] ——, "Multiple-antenna cooperative wireless systems: A diversity–multiplexing tradeoff perspective," *Information Theory, IEEE Transactions on*, vol. 53, no. 10, pp. 3371–3393, Oct. 2007.

[48] K. Azarian, H. E. Gamal, S. Member, and P. Schniter, "On the achievable diversity-multiplexing tradeoff in half-duplex cooperative channels," *IEEE Trans. Inform. Theory*, vol. 51, pp. 4152–4172, 2005.

[49] E. Stauffer, O. Oyman, R. Narasimhan, and A. Paulraj, "Finite-SNR diversity-multiplexing tradeoffs in fading relay channels," *Selected Areas in Communications, IEEE Journal on*, vol. 25, no. 2, pp. 245–257, February 2007.

[50] G. Kramer, M. Gastpar, and P. Gupta, "Cooperative strategies and capacity theorems for relay networks," *IEEE Transactions on Information Theory*, vol. 51, no. 9, pp. 3037–3063, Sept. 2005.

[51] A. Host-Madsen and J. Zhang, "Capacity bounds and power allocation for wireless relay channels," *Information Theory, IEEE Transactions on*, vol. 51, no. 6, pp. 2020–2040, 2005.

[52] A. Goldsmith, *Wireless Communications*. Cambridge University Press, 2005.

[53] G. Kramer, "Topics in multi-user information theory," in *Foundations and Trends in Communications and Information Theory*. Now Publishers, 2008, vol. 4, no. 4–5, pp. 265–444.

[54] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley and Sons, Inc., 1991.

[55] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Transactions on Information Theory*, vol. 22, no. 1, pp. 1–10, Jan. 1976.

[56] Z. Liu, V. Stankovic, and Z. Xiong, "Wyner-Ziv coding for the half-duplex relay channel," *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*, vol. 5, pp. 1113–1116, 18-23 March 2005.

[57] A. Chakrabarti, A. de Baynast, A. Sabharwal, and B. Aazhang, "Half-duplex estimate-and-forward relaying: Bounds and code design," *Information Theory, 2006 IEEE International Symposium on*, pp. 1239–1243, July 2006.

[58] J. Goblick, T., "Theoretical limitations on the transmission of data from analog sources," *Information Theory, IEEE Transactions on*, vol. 11, no. 4, pp. 558–567, Oct 1965.

[59] G. Kramer, I. Marić, and R. D. Yates, "Cooperative communications," *Found. Trends Netw.*, vol. 1, no. 3, pp. 271–425, 2006.

[60] T. E. Hunter and A. Nosratinia, "Diversity through coded cooperation," *Wireless Communications, IEEE Transactions on*, vol. 5, no. 2, pp. 283–289, 2006.

[61] A. Stefanov and E. Erkip, "Cooperative coding for wireless networks," *Communications, IEEE Transactions on*, vol. 52, no. 9, pp. 1470–1476, Sept. 2004.

[62] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *Communications, IEEE Transactions on*, vol. 36, no. 4, pp. 389–400, Apr 1988.

[63] M. Valenti and B. Zhao, "Distributed turbo codes: towards the capacity of the relay channel," *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, vol. 1, pp. 322–326 Vol.1, 6-9 Oct. 2003.

[64] M. Janani, A. Hedayat, T. Hunter, and A. Nosratinia, "Coded cooperation in wireless communications: Space-time transmission and iterative decoding," *IEEE Transactions on Signal Processing*, vol. 52, no. 2, pp. 362 – 371, Feb. 2004.

[65] D. Duyck, J. J. Boutros, and M. Moeneclaey, "Low-density graph codes for slow fading relay channels," Mar 2009. [Online]. Available: http://arxiv.org/abs/0903.1502

[66] Z. Zhang and T. Duman, "Capacity-approaching turbo coding and iterative decoding for relay channels," *Communications, IEEE Transactions on*, vol. 53, no. 11, pp. 1895–1905, Nov. 2005.

[67] ——, "Capacity-approaching turbo coding for half-duplex relaying," *Communications, IEEE Transactions on*, vol. 55, no. 10, pp. 1895–1906, Oct. 2007.

[68] A. Chakrabarti, A. D. Baynast, A. Sabharwal, and B. Aazhang, "Low density parity check codes for the relay channel," *Selected Areas in Communications, IEEE Journal on*, vol. 25, no. 2, pp. 280–291, February 2007.

[69] A. Chakrabarti, E. Erkip, A. Sabharwal, and B. Aazhang, "Code designs for cooperative communication," *Signal Processing Magazine, IEEE*, vol. 24, no. 5, pp. 16–26, Sept. 2007.

[70] G. Kramer, "Distributed and layered codes for relaying," *Signals, Systems and Computers, 2005. Conference Record of the Thirty-Ninth Asilomar Conference on*, pp. 1752–1756, October 28 - November 1, 2005.

[71] C. Li, M. Khojastepour, G. Yue, X. Wang, and M. Madihian, "LDPC code design for half-duplex relay networks," *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 2, pp. II–873–II–876, 15-20 April 2007.

[72] P. Razaghi and W. Yu, "Bilayer LDPC codes for the relay channel," *Communications, 2006. ICC '06. IEEE International Conference on*, vol. 4, pp. 1574–1579, June 2006.

[73] C. Hausl and J. Hagenauer, "Iterative network and channel decoding for the two-way relay channel," *Communications, 2006. ICC '06. IEEE International Conference on*, vol. 4, pp. 1568–1573, June 2006.

[74] G. Zeitler, R. Koetter, G. Bauch, and J. Widmer, "Design of network coding functions in multihop relay networks," *Turbo Codes and Related Topics, 2008 5th International Symposium on*, pp. 249–254, Sept. 2008.

[75] R. Hu and J. Li, "Practical Compress-Forward in user cooperation: Wyner-Ziv cooperation," *Information Theory, 2006 IEEE International Symposium on*, pp. 489–493, July 2006.

[76] Z. Liu, M. Uppal, V. Stankovic, and Z. Xiong, "Compress-Forward coding with BPSK modulation for the half-duplex Gaussian relay channel," *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*, pp. 2395–2399, July 2008.

[77] Y. Li, B. Vucetic, T. F. Wong, and M. Dohler, "Distributed turbo coding with soft information relaying in multihop relay networks," *Selected Areas in Communications, IEEE Journal on*, vol. 24, no. 11, pp. 2040–2050, 2006.

[78] W. Peterson and D. Brown, "Cyclic codes for error detection," *Proceedings of the IRE*, vol. 49, no. 1, pp. 228–235, Jan. 1961.

[79] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, pp. 471–480, Jul. 1973.

[80] R. J. Barron, "Systematic hybrid analog/digital signal coding," Ph.D. dissertation, MIT, USA, 2000.

[81] S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (DISCUS): design and construction," *IEEE Transactions on Information Theory*, vol. 49, no. 3, pp. 626–643, Mar. 2003.

[82] M. Hellman, "Convolutional source encoding," *IEEE Transactions on Information Theory*, vol. 21, no. 6, pp. 651–656, Nov. 1975.

[83] V. Koshelev, "Direct sequential encoding and decoding for discrete sources," *IEEE Transactions on Information Theory*, vol. 19, no. 3, pp. 340–343, May 1973.

[84] G.-C. Zhu, F. Alajaji, J. Bajcsy, and P. Mitran, "Transmission of nonuniform memoryless sources via nonsystematic turbo codes," *Communications, IEEE Transactions on*, vol. 52, no. 8, pp. 1344–1354, Aug. 2004.

[85] X. Jaspar, "Joint source-channel turbo techniques and variable length codes," Ph.D. dissertation, UCL, Belgium, 2008.