Code-Aided Synchronization for Digital Burst Communications

Cédric Herzet

Communications Laboratory,
Université catholique de Louvain,
Pl. du Levant 2, B1348 Louvain-la-Neuve, Belgium

18th April 2006

Esther demande pourquoi les gens sont tristes.

"C'est simple, répond le vieillard. Ils sont prisonniers de leur histoire personnelle. Tout le monde est convaincu que le but de cette vie est de suivre un plan. Personne ne se demande si ce plan est le sien ou a été inventé par quelqu'un d'autre. Tous accumulent des expériences, des souvenirs, des objets, des idées qui ne sont pas les leurs et c'est plus qu'ils ne peuvent porter. Et c'est ainsi qu'ils oublient leurs rêves."

Paulo Coelho, Le Zahir, 2005.

Remerciements

Après plusieurs années de travail, me voici enfin arrivé à l'épreuve ultime qui consiste à rédiger la partie "remerciements" de ma thèse... Au moment où j'écris ces lignes, quelques jours seulement me séparent de ma présentation publique, présentation à l'issue de laquelle je serai (en principe...) proclamé docteur. La remise de cette thèse me force à me retourner, à scruter loin derrière moi, me demandant ce qu'aurait pu être ma vie sans avoir fait les mêmes choix mais surtout, surtout, sans avoir eu la chance de rencontrer les mêmes personnes. Sans doute bien différente... Ces quatre dernières années, il m'a été donné de faire de ce que j'aimais: apprendre et créer. Cette thèse, bien plus que l'oeuvre d'une seule personne, est l'oeuvre de tout ceux qui m'ont entourés et accompagnés jusqu'ici. Par ces quelques lignes, j'aimerais les remercier.

Premièrement, je voudrais remercier ma famille et plus particulièrement mes parents. Je leur dédie ce travail. Ils m'ont toujours aidé et guidé tout en respectant mes choix. Ils m'ont appris, par leur exemple, le respect de soi et de chacun. Ils m'ont enseigné la remise en question critique de toutes choses (ce qui s'avéra bien utile ces quatre dernières années...) A vous, de tout coeur, merci !

J'aimerais ensuite remercier les personnes qui ont fait que ce travail fût pour moi plus une quête passionnée qu'un objectif laborieux. Bien sûr, tout passionné possède ses maîtres; ces personnes, qui plus que vous enseigner une matière, vous transmettent une passion. Mr Defauwe, mon "vieux" prof de math fût sûrement un de ceux-là. Oeuvrant sans relâche (mercredi après-midi et samedi matin compris!), absorbant les frustrations de ses pauvres élèves et vivant leurs victoires, il fut sans aucun doute un moteur pour moi à de nombreux moments et certainement la personne m'ayant

permis d'avoir une "autre" vision de nombreux problèmes mathématiques auxquels j'ai été amené à faire face bien plus tard...

Ensuite, vînt Mr Vandendorpe, mon professeur de traitement de signal et de télécommunication. Sa rigueur et sa pédagogie, choses m'ayant bien souvent cruellement manqué jusqu'alors dans ce système universitaire, me séduirent assez rapidement... Toujours à la tâche, démontrant minutieusement chaque résultat, il eut également à coeur de nous faire comprendre la signification des équations qu'il nous enseignait, nous ouvrant une porte vers de nouvelles dimensions... Tant et si bien que, quelques années plus tard, ce qui devait arriver arriva: Mr Vandendorpe devînt Luc, mon promoteur de thèse. Tâche ingrate en vérité que celle de promoteur... tâche qu'il parvînt toutefois à relever avec brio. Je tiens sincèrement à le remercier pour la confiance qu'il a placé en moi tout au long de ces années et pour le soutien qu'il m'a apporté.

La qualité d'une thèse n'est pas seulement due à un ensemble de matières que l'on assimile mais aussi, et surtout, aux gens qui vous entourent. Ceux-là même qui font que vous vous sentiez bien, qui vous comprennent, qui vous inspirent... Aussi je ne pourrais que trop remercier mes vieux compagnons de route, Manu et Enguerrand, ayant suivi (et enduré) mes frasques universitaires et bien d'autres choses encore... Toujours présents, dans les moments difficiles comme dans les moments de joie, ils m'ont souvent apporté plus d'inspiration qu'ils ne l'imaginent...

Mes très chers collègues de bureau méritent eux-aussi une mention spéciale. Passer plus de quatre ans dans le même bureau que moi n'a pas du être tous les jours de tout repos. Xavier, Valéry, Jacek et Raph, merci à vous! J'ai réellement apprécié ces quatre années passées à travailler (et discuter) à vos cotés.

Je ne pourrais bien évidemment pas oublier toutes les autres personnes du labo, qui sont pour certaines devenues plus des amis que des collègues. Merci à toi Rosita! Merci pour ton aide et tes encouragements de tous les instants. Merci pour ta vision si humaine du monde et des choses. Tu fus souvent un oasis de réconfort dans ce monde de compétition parfois bien trop inhumain à mon goût. Merci à vous, Jon et Harold, pour les fabuleuses soirées "taquet" que nous avons passées ensemble. Je ne suis pas prêt de les

oublier... Merci à tous les autres, Xa, Bert, Lya, Ana... pour votre gentillesse et votre bonne humeur de tous les instants. Tous ensemble, vous avez fait que rentrer chaque matin dans ce labo soit un plaisir plutôt qu'un devoir.

Enfin, merci à tous les autres: Sylvain, Sabine, Cécile, Thibauld, Milou, Co, Didj, Alex, Mitch, Nath, Kimi, Claire... et ceux que je ne peux citer ici par souci du lecteur (qui déjà trépigne de pouvoir la suite de cette thèse) mais qui ont participé à faire de ces quatre années (et plus) une réussite. Merci à tous!

Abstract

This thesis deals with the synchronization of digital communication systems. Synchronization (from the Greek syn (together) and chronos (time)) denotes the task of making two systems running at the same time. In communication systems, the synchronization of the transmitter and the receiver requires to accurately estimate a number of parameters such as the carrier frequency and phase offsets, the timing epoch...

In the early days of digital communications, synchronizers used to operate in either data-aided (DA) or non-data-aided (NDA) modes. However, with the recent advent of powerful coding techniques, these conventional synchronization modes have been shown to be unable to properly synchronize state-of-the-art receivers.

In this context, we investigate in this thesis a new family of synchronizers referred to as code-aided (CA) synchronizers. The idea behind CA synchronization is to take benefit from the structure of the code used to protect the data to improve the estimation quality achieved by the synchronizers. In a first part of the thesis, we address the issue of turbo synchronization, i.e., the iterative synchronization of continuous parameters. In particular, we derive several mathematical frameworks enabling a systematic derivation of turbo synchronizers and a deeper understanding of their behavior. In a second part, we focus on the so-called CA hypothesis testing problem. More particularly, we derive optimal solutions to deal with this problem and propose efficient implementations of the proposed algorithms. Finally, in a last part of this thesis, we derive theoretical lower bounds on the performance of turbo synchronizers.

Contents

Table of contents viii

Notations and Glossary xii

1 Introduction 1

2 System Model 11
2.1 Digital-Communication Systems 11
2.2 Bandpass Linearly-modulated Signal Transmission 14

3 Estimation Theory 19
3.1 Basic Principles 19
3.2 Minimum-Mean-Square-Error Estimation 21
3.3 Maximum-A-Posteriori Estimation 22
3.4 Maximum-Likelihood Estimation 22
3.5 Estimators Constrained to a Particular Mathematical Structure . 23
3.5.1 Linear MMSE Estimator 24
3.5.2 Best Linear Unbiased Estimator 24
3.6 Sufficient Statistics 25
3.7 Assessment of the Estimator Performance 26
3.7.1 The Bayesian Cramer-Rao Bound 27
3.7.2 The Standard Cramer-Rao Bound 28
3.7.3 The Modified Cramer-Rao Bound 30
3.8 Main Points 31

4 Optimal Reception and Synchronization 33
4.1 Sequence-wise Optimal Reception 33
4.2 Bit-wise Optimal Reception 34
4.3 Synchronization: A Suboptimal Approach 35

4.4 Synchronization: Principle . . . 37
4.5 Synchronization Modes . . . 38
4.6 Why Code-aided Synchronization ? . . . 41
4.6.1 Continuous-parameter Estimation . . . 41
4.6.2 Discrete-parameter Estimation . . . 43
4.7 Main Points . . . 45

5 Factor Graphs and Sum-Product Algorithm 47
5.1 Marginal Computation: Problem Statement . . . 47
5.2 The Distributive Law and the Factor Graph Representation . . . 48
5.3 The Sum-Product Algorithm . . . 50
5.4 Application of the SP Algorithm to Receiver Tasks . . . 52
5.4.1 MAP Sequence Detection . . . 53
5.4.2 MAP Bit Detection . . . 55
5.5 Main Points . . . 65
5.6 Appendix . . . 66

6 Iterative Optimization Algorithms 69
6.1 Convergence of Iterative Descent Methods . . . 69
6.1.1 Global Convergence Theorem . . . 70
6.1.2 Speed of Convergence . . . 71
6.2 Gradient Methods . . . 71
6.2.1 The Steepest-Descent Algorithm . . . 73
6.2.2 The Newton-Raphson Algorithm . . . 73
6.3 EM-based Algorithms . . . 74
6.3.1 The Expectation-Maximization Algorithm . . . 74
6.3.2 The Generalized EM Algorithm . . . 76
6.4 Main Points . . . 77
6.5 Appendix . . . 78

7 Code-Aided Synchronization of Continuous Parameters 81
7.1 ML Synchronization . . . 82
7.2 Conventional Suboptimal Approaches . . . 83
7.2.1 Data-Aided Synchronizers . . . 83
7.2.2 Non-Data-Aided Synchronizers . . . 84
7.3 Implementation Modes . . . 86
7.4 Iterative Synchronization Based on the EM Algorithm . . . 91
7.4.1 The EM Algorithm Applied to ML Synchronization . . . 91
7.4.2 The ECM Algorithm Applied to ML Synchronization . . . 94

7.4.3 Implementation of the EM-based Synchronizer 94
7.5 Iterative Synchronization Based on Gradient Methods 98
7.5.1 The SD algorithm Applied to ML Synchronization . . . 99
7.5.2 Implementation of the SD Synchronizer 100
7.5.3 The NR Algorithm Applied to ML Synchronization . . . 103
7.5.4 Implementation of the NR Synchronizer 104
7.6 Combining the Gradient and the EM Synchronization Frameworks 107
7.7 The SP Algorithm: a General Framework for Iterative Synchronization 108
7.7.1 The SP-algorithm Applied to Bit-wise Optimal Reception 109
7.7.2 Message Approximations and Iterative Synchronization 111
7.7.3 ML Update of the SP Estimates 114
7.7.4 Fixed Points of the SP Synchronizer 115
7.7.5 Combining the SP Algorithm and the EM Algorithm . . 116
7.7.6 Combining the SP Algorithm and Gradient Algorithms 121
7.8 Main Points 123
7.9 Appendices 124
7.9.1 Proof of (7.41) 124
7.9.2 Proof of (7.49) 125
7.9.3 Proof of (7.77) 126
7.9.4 Proof of (7.93) 127

8 Code-Aided Synchronization of Discrete Parameters 129
8.1 Conventional Approaches 130
8.2 MAP Synchronization 131
8.2.1 Data-oriented MAP-based Synchronization 132
8.2.2 MAP synchronization via the SP algorithm 135
8.2.3 Sequential MAP Synchronization on Observation Subsets 140
8.2.4 Implementation and Complexity of the MAP Synchronizers 143
8.3 Data-oriented Synchronization 145
8.3.1 FER-wise Optimal Synchronization 146
8.3.2 The BER-wise Optimal Receiver Revisited 150
8.3.3 The BER-wise Optimal Synchronizer 152
8.4 Main Points 153
8.5 Appendices 154

8.5.1 Proof of (8.9) 154

9 Lower Bounds on the Synchronizer Performance 157
9.1 Lower Bounds on the MSE 157
9.1.1 Cramer-Rao Bound in the Presence of a Deterministic Soft-information Vector 158
9.1.2 Lower Bounds on the MSE in the Presence of a Random Soft-information Vector 159
9.1.3 Practical Evaluation of the SCRB and the SMCRB 162
9.2 Lower bound on the speed of convergence 162
9.3 Main Points 165
9.4 Appendices 166
9.4.1 Proof that $\mathrm{SMCRB}(\mathbf{b}) \geq \mathrm{MCRB}(\mathbf{b})$ 166
9.4.2 Proof that $\lim_{K\to\infty} \mathrm{SCRB}(\mathbf{b}) = \mathrm{SMCRB}(\mathbf{b})$ 168
9.4.3 Proof of (9.24) 169

10 Simulation Results 173
10.1 Lower Bounds for Soft-information-directed Synchronizers . . . 174
10.1.1 Theoretical Soft-information Distribution 174
10.1.2 Soft-information Delivered by a Turbo Receiver 178
10.2 Performance of Continuous-parameter Synchronizers 180
10.2.1 Synchronizer Performance: Speed of Convergence . . . 182
10.2.2 Synchronizer Performance: Ultimate Accuracy 195
10.2.3 Synchronizer Performance: Acquisition Range 200
10.2.4 Synchronization of State-of-the-art Receivers 204
10.3 Performance of Discrete-parameter Synchronizers 221
10.3.1 Detection Ability of CA Synchronizers 222
10.3.2 Synchronization of a Convolutionally-coded Transmission 225

11 Conclusions 229

Bibliography 239

Notations and Glossary

x^*	the complex conjugate of x
$\mathrm{Re}\{x\}$	the real part of x
$\mathrm{Im}\{x\}$	the imaginary part of x
$\|x\|$	the amplitude of x
$\arg\{x\}$	the argument of x
$\hat{x}$	an estimate of x
$\mathbf{x}$	a vector $\mathbf{x}$
$\mathbf{x}^T$	the transpose of vector $\mathbf{x}$
x_k	the k-th element of vector $\mathbf{x}$
$\mathcal{X}$	a set of elements
$\|\mathcal{X}\|$	the number of elements in $\mathcal{X}$
$\mathbf{X}$	a matrix $\mathbf{X}$
$\mathbf{X}^T$	the transpose of matrix $\mathbf{X}$
$\mathbf{X}^H$	the transpose complex conjugate of matrix $\mathbf{X}$
$\mathbf{X}^{-1}$	the inverse of matrix $\mathbf{X}$
$\mathbf{X}_{ij}$	the element at row i and column j of matrix $\mathbf{X}$
$\mathbf{I}_{N\times N}$	$N \times N$ identity matrix
$x \sim f(x)$	x is distributed according to $f(x)$
$x \propto y$	x is equal to y up to irrelevant additive and multiplicative terms
$x \simeq y$	x is approximatively equal to y
$x \triangleq y$	x is defined as y
$x \ll y$	x is much less than y
$\mathbf{X} \succ \mathbf{Y}$	$\mathbf{X} - \mathbf{Y}$ is positive definite
$\mathbf{X} \succeq \mathbf{Y}$	$\mathbf{X} - \mathbf{Y}$ is positive semi-definite
$\oplus$	addition operation in the considered commutative semi-ring
$\odot$	multiplication operation in the considered commutative semi-ring
$\otimes$	convolution operation
$\sum_{\sim\{x\}}$	summation over all the variables but x

$\sum^{\oplus}$	summation using addition $\oplus$
$\prod^{\otimes}$	product using multiplication $\otimes$
$\arg\max_x\ f(x)$	the value of x that maximizes $f(x)$
$\nabla f(x)$	the gradient of $f(x)$
$\nabla^2 f(x)$	the Hessian of $f(x)$
$\mu_{x\to y}(y)$	the sum-product message going from node x to node y.
$\delta(\cdot)$	the Dirac function
$\mathbb{I}\{\cdot\}$	the indicator function
$\mathrm{erfc}(\cdot)$	the complementary error function
$E_x[f(x)]$	the expectation operator: $E_x[f(x)] = \int f(x)\, p(x)\, dx$
$E_{x\|y}[f(x)]$	the conditional expectation operator: $E_{x\|y}[f(x)] = \int f(x)\, p(x\|y)\, dx$
$\mathbf{m_x}$	the mean value of $\mathbf{x}$: $\mathbf{m_x} = E_{\mathbf{x}}[\mathbf{x}]$
$\mathbf{P_{xx}}$	the covariance matrix of $\mathbf{x}$: $\mathbf{P_{xx}} = E_{\mathbf{x}}[\mathbf{x}^T\mathbf{x}] - \mathbf{m}_{\mathbf{x}}^T\mathbf{m_x}$
$\mathbf{P_{xy}}$	the cross-covariance matrix of $\mathbf{x}$ and $\mathbf{y}$: $\mathbf{P_{xy}} = E_{\mathbf{x}}[\mathbf{x}^T\mathbf{y}] - \mathbf{m}_{\mathbf{x}}^T\mathbf{m_y}$
$\mathcal{N}(\mathbf{m_x}, \mathbf{P_{xx}})$	the normal distribution with mean $\mathbf{m_x}$ and covariance $\mathbf{P_{xx}}$.

$\mathbf{u}$	the vector of information bits
$\mathbf{x}$	the vector of coded bits
$\mathbf{a}$	the vector of transmitted symbols
$\mathbf{r}$	the vector of observations
$\mathbf{y}$	the vector of matched-filter outputs
$\mathbf{w}$	the vector of white Gaussian noise samples
$\mathbf{b}$	the vector of synchronization parameters
A	the real channel gain
ν	the carrier frequency offset
θ	the carrier phase offset: $\theta = k_\theta\Psi + \epsilon_\theta$
k_θ	the phase ambiguity
ϵ_θ	the fractional phase offset
Ψ	the rotational symmetry of the constellation
τ	the timing offset: $\tau = k_\tau T + \epsilon_\tau$
k_τ	the timing ambiguity
ϵ_τ	the fractional timing offset
T	the symbol period
T_s	the sampling period
L	the number of information bits per frame
K	the number of symbols per frame
$\mathcal{U}$	the set of possible information-bit sequences

$\mathcal{C}$	the set of possible coded-bit sequences
$\mathcal{A}$	the signaling constellation
$\mathcal{A}^K$	the set of possible symbol sequences
$\mathcal{B}$	the set of the possible values for the synchronization parameters
$p(t)$	the transmit pulse
$s(t)$	the transmit signal
$r(t)$	the received signal
$w(t)$	the additive white Gaussian noise
$p_e(a_k)$	the extrinsic probability of symbol a_k
$\chi(\cdot)$	the coding function
$\varphi(\cdot)$	the mapping function

Introduction

1

In any communication system, some transmitter wishes to send some information to a receiver. When considering digital communications, the information to transmit takes the form of binary values, usually referred to as *bits*. The purpose of digital-communication theory consists in designing transmission systems as well as receiving structures which ensure reliable transmissions. In this context the question may arise of the ultimate amount of information one can reliably transmit over some channel. The answer to this question was brought by Claude Shannon in 1948 [1]. Shannon basically derived a higher bound on the transmission rate, referred to as *capacity*, beyond which no information can be transmitted without errors. On the contrary, he proved that, as long as a proper code is used, arbitrary reliability can be achieved for information rates lower than the capacity.

For that moment on, digital communication community has been looking for practical coding structures enabling to operate close to the capacity limit. Indeed, although Shannon's theory proved the existence of such codes, it did not give any clue to design them. In fact, Shannon's theory even did not prove the existence of capacity-approaching codes with tractable decoding complexity. In 1993, Claude Berrou's research group proposed a coding structure, referred to as *turbo code* [2], operating close to Shannon's bound while exhibiting reasonable decoding complexity. Perhaps more than the proposed coding structure, the most groundbreaking contribution of Berrou's team lay in the iterative processing used for decoding the received observations. The so-called *turbo decoder* consisted in two low-complexity decoders iteratively exchanging some soft information about the transmitted bits. Due to its outstanding performance when applied to the decoding of

turbo codes, the so-named *turbo principle* has then been further applied to a variety of other receiver tasks: demodulation [3], equalization [4], multi-user joint reception [5]... Note that, although leading to remarkable performance, the turbo principle was originally developed in a rather ad-hoc way. Only recently a mathematical framework was proposed confirming, up to a point, the validity of these algorithms. This framework goes under the name of *"factor graphs"* [6, 7] and provides insightful ways to develop iterative algorithms, based on a graphical representation of a problem or system.

Apart from the transmitted information bits, the signal to be processed by the receiver depends on a variety of other unknown parameters. These ones, often referred to as *nuisance* parameters, include the channel impulse response, the channel propagation delay, the carrier phase offset, the carrier frequency offset... Most of the state-of-the-art receivers cited above exclusively focus on the estimation of the information bits and assume that the other parameters are perfectly known at the receiver side. In practice however, the uncertainty associated to nuisance parameters has to be taken into account by the receiver. In order to do so, the most common approach consists in first computing an estimate of the nuisance parameters and then performing bit detection/decoding by considering the estimate as the actual parameter value. When applied to the estimation of the carrier phase offset, frequency offset or timing epoch, this approach is commonly referred to as *synchronization*.

During the last decades, synchronization was not a major source of degradations. Indeed, communication systems usually operated at fairly high signal-to-noise ratios (SNRs) and existing algorithms [8, 9] enabled to properly synchronize the system. However, with the recent discovery of powerful error-correcting codes [2, 10, 11] and iterative decoding techniques [2, 4, 5], communication systems are nowadays able to operate at very low signal-to-noise ratios. As a consequence, conventional synchronization algorithms [8, 9], developed between the 1960s and 1990s to operate at fairly high SNRs, can no longer be successfully applied. New synchronization algorithms delivering very accurate parameter estimates while operating at very low SNRs are required to properly synchronize state-of-the-art receivers. The synchronization task, which was previously considered as of secondary interest, has therefore become of major importance in the design of communication systems.

At very low SNRs, it has been shown (see [12], [13], [14]) that taking the code structure into account in the synchronization task enables to significantly improve the parameter estimation quality. Unlike conventional synchronization methods [8, 9], state-of-the-art synchronization algorithms should therefore take benefit from the sequence a priori information available from the code structure. This approach has been referred to as *code-aided* (CA) synchronization in contrast to conventional data-aided (DA) and non-data-aided (NDA) synchronization modes which disregard the code structure in the estimation process. Unfortunately, the derivation of practical code-aided synchronizers is not a straightforward task. An idea proposed by several authors to help the synchronizer to improve the estimation quality was to use some symbol information coming from the data detection part of the receiver. The justification of this approach rests on the observation that the gap between conventional (i.e. DA and NDA) and CA synchronizer performance is due to lack of information about the *data* symbols in the former case. Taking benefit in the synchronizer from some symbol information delivered by the symbol-detection part of the receiver, we can expect to get closer to the CA performance and therefore improve the estimation quality.

Since the advent of turbo-codes in the early 1990s, most state-of-the-art receivers have been handling *soft information* about the transmitted data. Soft information, by opposition to hard information, keeps a track of the confidence about the decision made on the transmitted data. New synchronization algorithms (see e.g. [15, 16, 17, 18]) have therefore been designed to take benefit from this symbol soft information available in the receivers. However, most of the proposed estimation algorithms were developed in an ad-hoc way and are therefore only useful for very specific applications. Mathematical frameworks enabling to predict synchronizer performance and derive synchronizers in a systematic way were still missing.

The purpose of this work is to derive mathematical tools for code-aided synchronization[1]. In particular, we intend to answer the following questions:

1. What kind of soft information should be handled by the CA synchronizer?

[1] We refer the interested reader to Wymeersch's [19] and Dauwels [20] thesis which have tackled the same kind of problems.

2. How should be used the soft information to lead to efficient synchronizers?

3. What are the properties of the synchronizers handling soft information? In particular, if the synchronizers are iterative, what are the final convergence points? What is the speed of convergence of these synchronizers?

Note that this dissertation does not focus on the design and the optimization of specific communication systems. Instead, it rather intends to derive mathematical tools enabling to answer the above questions. For the sake of example, the mathematical frameworks derived in this dissertation will be illustrated by the synchronization of a bandpass linearly-modulated signal [21]. This simple example will allow us to clearly expose the ideas motivating the proposed approaches and derive simple and meaningful expressions of the synchronizers. The methods proposed in this thesis are however quite general and can be straightforwardly extended to a variety of more complex communication schemes.

How to read this thesis ?

A deep understanding of the synchronization mathematical frameworks derived in this dissertation requires a number of background knowledges. In particular, the reader should be familiar with communication theory, estimation theory, optimization theory and factor-graph representation theory. Being aware that not all the readers have a full understanding of all these topics, this dissertation includes some chapters dealing with some background information. These chapters will allow us to recall or emphasize some notions the reader should be aware of to fully understand the remainder of the dissertation. As a consequence, not all the chapters are relevant for all the readers: some chapters should be skipped by those having a sufficient knowledge in the corresponding field. To help the reader skip chapters, Fig. 1.1 represents a dependence graph of the sections: the material of a chapter should be understood to fully understand the chapter which is connected to it by an arrow. For example, the material related to estimation theory in chapter 3 should be understood before moving on to chapter 4. Each chapter of the dissertation ends with a section entitled ”Main Points” which lists the elements the reader should be aware of after having gone through the chapter.

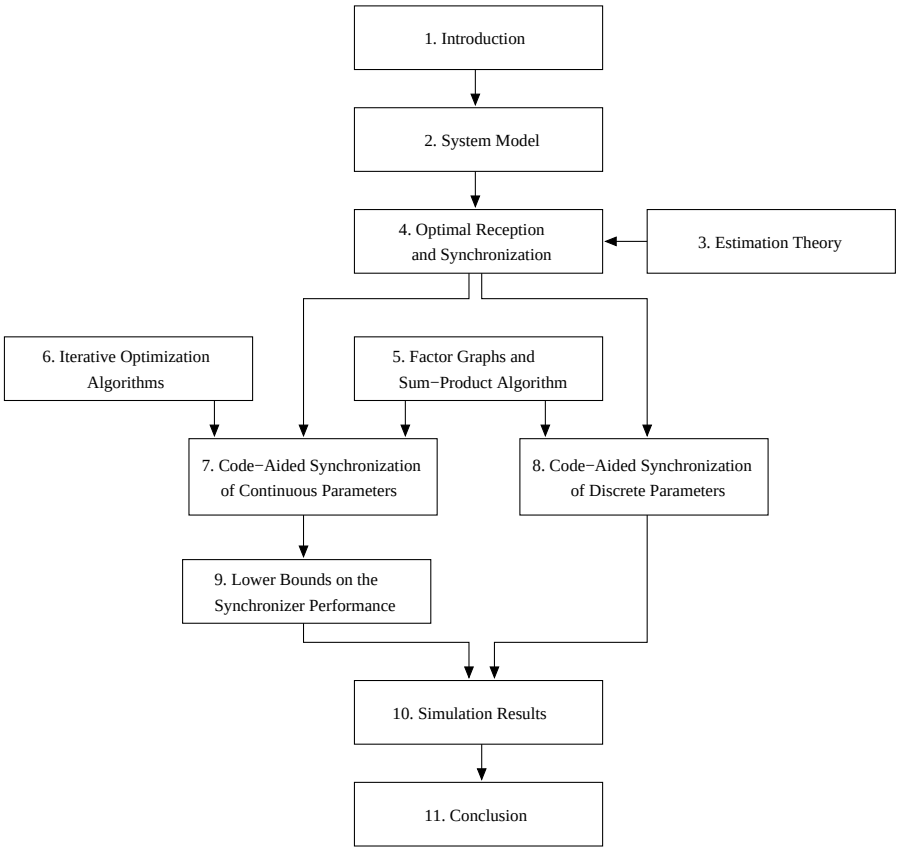

Figure 1.1 Chapter dependence graph.

The personal contributions of this work are included in chapters 7, 8, 9 and 10. The synchronization issues are divided into the estimation of continuous and discrete-valued parameters. Chapter 7 deals with the synchronization of continuous parameters. Chapter 8 copes with the estimation of discrete parameters. In chapter 9, we derive lower bounds for assessing the performance of soft-information-directed synchronizers. Finally, chapter 10 illustrates the performance of the synchronization algorithms derived in the previous chapter by means of simulation results. Mathematical details are usually placed in the appendices of each chapter. This way, the readers can get through the chapters without getting distracted by the mathematical details.

Outline

The dissertation is divided into 11 chapters, including this introduction and the conclusion. We give now a short overview of each of them.

Chapter 2: System Model

In this chapter we introduce the model that will be used for the derivation of synchronization algorithms in the remainder of this thesis. In a first part, we will describe the different parts making up a common digital-communication scheme. We then introduce the expressions of the transmitted and received signals in the case of a bandpass linearly-modulated transmission scheme. In particular, we emphasize some of the synchronization issues which may appear in such a case.

Chapter 3: Estimation Theory

In this chapter we recall the fundamental principles of estimation theory. In particular, the main estimators are reviewed and their main characteristics emphasized. We then introduce the concept of *sufficient statistics* or, in other words, we answer the question "When is a data set equivalent to another one regarding the estimation of a parameter?" Finally, the question of the ultimate accuracy of any estimator is tackled. In particular, we present the main expressions of both the Cramer-Rao bound and the modified Cramer-Rao bound.

Chapter 4: Optimal Reception and Synchronization

In this chapter, we derive the general expressions of the sequence-wise and bit-wise optimal receivers. We then introduce the concept of *synchronization* and we place it in the context of the optimal reception. It is emphasized that synchronization may be performed under different *synchronization modes*: data-aided (DA), non-data-aided (NDA) and code-aided (CA). The latter mode is shown to be the only one of interest in state-of-the-art receivers and will therefore be the purpose of the remainder of the dissertation.

Chapter 5: Factor Graphs and Sum-Product Algorithm

In this chapter, we introduce the reader to factor-graph representation and its associated "sum-product" algorithm. General principles are first presented

and shown to justify some powerful existing receiving schemes. In particular, we show that well-known algorithms such as the Viterbi decoder, the BCJR decoder, the turbo decoder and the turbo demodulator previously proposed in the literature are only particular instances of the SP algorithm. This chapter is primordial to understand the nature of the soft information which will be handled by the synchronizers proposed in the subsequent chapters.

Chapter 6: Iterative Optimization Algorithms

In this chapter, we consider iterative procedures enabling to solve unconstrained optimization problems. The questions of the final convergence points and the speed of convergence of iterative algorithms are addressed. In particular, we state the *Global Convergence Theorem* which gives sufficient conditions for the convergence of an algorithm to a point of a given solution set. We then present two well-known families of iterative optimization procedures, namely the families of gradient and expectation-maximization algorithms. The properties of these algorithms are presented and discussed.

Chapter 7: Code-Aided Synchronization of Continuous Parameters

In this chapter, we address the problem of maximum-likelihood (ML) code-aided (CA) synchronization of continuous parameters. Three mathematical frameworks are derived for the design of ML CA iterative synchronizers. These are shown to give a justification to several synchronization methods proposed in the literature.

The first synchronization framework is based on the expectation-maximization (EM) algorithm. The equations of the EM-based synchronizer are derived in the particular case of a bandpass linearly-modulated signal. The implementation and the complexity of this kind of synchronizer is then discussed and practical approximations are proposed.

The second synchronization framework is based on gradient algorithms. We consider two particular gradient algorithms, namely the steepest-descent (SD) and the Newton-Raphson (NR) algorithms. We derive expressions enabling to efficiently evaluate the gradient and the Hessian matrix of the log-likelihood function which are required for the implementation of this

kind of synchronizers. The implementation and the complexity of these algorithms are discussed and compared with the EM algorithm. Practical approximations are proposed to simplify their implementation.

The third synchronization framework rests on the sum-product (SP) algorithm. Unlike the EM and the gradient algorithms, the properties of this algorithm are not well established yet. We therefore discuss the fixed points of the synchronizer derived from the proposed SP synchronization framework. In particular, we prove that, like the EM and the gradient algorithms, the fixed points of the SP synchronizer are the stationary points of the likelihood function. We then combine the frameworks based on the EM and the gradient algorithms with the SP synchronization framework. Doing this, we show that the synchronizer we earlier derived from some practical approximations of the EM (resp. the gradient) algorithm is only a particular approximation of a general framework combining the SP and the EM (resp. gradient) algorithms.

Chapter 8: Code-Aided Synchronization of Discrete Parameters

In this chapter, we address the problem of code-aided (CA) synchronization of discrete-valued parameters. In particular, we consider the problems of carrier-phase ambiguity resolution and frame synchronization. CA synchronization algorithms are derived according to two main approaches.

We first consider the maximum a posteriori (MAP) criterion. The design of the synchronizers are therefore placed in the context of the minimization of the probability of committing a wrong decision about the synchronization parameter. Two synchronizers are derived. The first one is a "data-detection-aware" approximation of the actual MAP synchronizer: we emphasize that this synchronizer is equivalent to a synchronizer recently proposed in the literature in an ad-hoc fashion. We then place the MAP synchronization problem in the context of the SP algorithm and the factor-graph representation. We show that, contrary to the common believe, the actual MAP solution may be implemented with a tractable complexity.

In a second part, we design synchronizers based on "data-oriented" criteria: we propose synchronizers based on the minimization of either the frame-error rate (FER) or the bit-error rate (BER) achieved by the communication

system.

Chapter 9: Lower Bounds on the Synchronizer Performance

In this chapter we derive lower bounds on the performance of ”soft-information-directed” synchronizers.

In a first part, we derive ”soft-information-aware” lower bounds on the synchronization mean square error (MSE). In other words, we intend to answer the following question: “Given an amount of soft information about the transmitted sequence, what is the best performance achievable by any (unbiased) soft-information-aided estimator ?” Two lower bounds, namely the soft Cramer-Rao bound (SCRB) and the soft modified Cramer-Rao bound (SMCRB), are considered and their properties discussed.

In a second part, we address the question of the speed of convergence of the EM synchronizer. In particular, we show that its speed of convergence may be lower-bounded by a function of the ratio between the Cramer-Rao bound and the modified Cramer-Rao bound associated to the estimation problem.

Chapter 10: Simulation Results

In this chapter we illustrate by simulation results the performance of the synchronizers derived in the previous chapters.

System Model

2

In this chapter we introduce the model that will be used for the derivation of synchronization algorithms in the remainder of this thesis. First, we will describe the different parts making up a common digital-communication scheme. Secondly, we will introduce the expressions of the transmitted and received signals in the case of a bandpass linearly-modulated transmission scheme. In particular, we emphasize some of the synchronization issues which may appear in such a case. Some parts of this chapter are inspired from [19, 20].

2.1 Digital-Communication Systems

During the last decade, the demand for high-data-rate and good-quality-of-service digital communications has grown in an explosive manner. This increasing demand has been calling for innovative techniques that improve both link reliability and system capacity while exhibiting fair implementation complexity. The intensive investigation performed by the coding community has led to the discovery of powerful error-correcting codes ([22],[10]) operating very close to the channel capacity. On top of that, new ways to communicate have been proposed. Important examples are multi-antenna (so-called MIMO) systems [23, 24], code-division multiple access (CMDA) transmissions [25, 26], orthogonal frequency division multiplexing (OFDM) [27, 28]... The combination of this plethora of communication techniques may lead to a very large number of configurations, sometimes quite different in appearance. These different digital-communication schemes may however be summarized via the scheme represented in Fig. 2.1. In this figure, we may distinguish between the following stages:

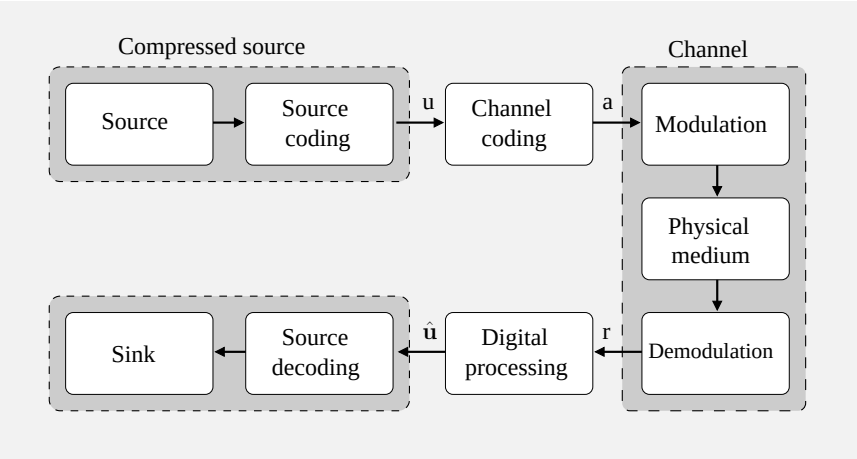

Figure 2.1 General communication scheme.

- **Source Coding:** The data to send may contain some redundancies. These redundancies reduce the system efficiency since some part of the information is transmitted several times. In practice, the data is therefore compressed to limit the quantity of useless information transmitted by the system. An ideal source compression should lead to independent information messages with uniform distribution [29].

- **Channel Coding:** Channel coding is somehow the reverse operation of source coding. It consists in introducing some structured redundancy among the transmitted data. The goal of this operation is to protect the transmitted data against the errors which may occur due to the noisy nature of the channel. Ideal channel codes should enable error-free transmissions as long as the system operates below the channel capacity [30]. In practice, codes exhibiting such a property while having reasonable decoding complexity are not easy to find and have been addressed a lot of research during the last 50 years. Only recently, some codes operating very close to the channel capacity while exhibiting reasonable decoding complexity have been discovered, e.g. see [22], [10].

- **Modulation/Physical medium/Demodulation:** After having undergone source and channel coding, the data has to be sent from the

transmitter to the receiver through a *physical medium*. Examples of physical media includes free space, wire lines (DSL), optical fiber cables... The propagation of the signal through a physical medium leads to degradations: the signal may be distorted due to the non-flat frequency response of the medium and corrupted by different kinds of additive noises [21].

In order to be transmitted through the physical medium, the data has to be converted into an analog form. The nature of the analog signal used for the transmission depends on the medium characteristics and the considered application. This digital-to-analog operation is usually referred to as *modulation*. The reverse operation is the *demodulation*. The latter consists in the transformation of the received analog signal to discrete-time samples which can be further processed by the digital receiver.

The modulation, physical medium and demodulation blocks characterize the relation between the transmitted coded symbols and the received observation samples. Their concatenation is usually referred to as *channel*. The features of the channel can be characterized by a set of parameters, which are usually unknown at the receiver.

- **Digital Processing:** In order to recover the transmitted data, the observation samples delivered by the demodulator are further processed by a digital-processing device. The role of the digital-processing device basically consists in computing the best possible estimate $\hat{\mathbf{u}}$ of the transmitted data. An ideal digital-processing device should minimize the number of errors committed on the decisions made about the transmitted data. Chapter 4 gives the general equations of the optimal receiver in that sense.

  In practice, the implementation of the optimal receiver is usually intractable and we have therefore to resort to suboptimal variants. The operations performed by the digital-processing device are then usually divided into different tasks such as equalization, demapping, decoding, synchronization... See [21] for more details about the classical ways to perform these operations.

- **Source decoding:** The final operation of the receiver consists in recov-

ering the data from estimate $\hat{\mathbf{u}}$. Note that this operation may sometimes be included in the "Digital Processing" part. In [31, 32] for example the authors propose an efficient method for joint channel and source decoding. In this work, we will however assume that the source-decoding device is independent from the digital-processing part.

In this work, we will focus on the implementation of efficient synchronization algorithms which can be embedded in the digital-processing part of the receivers.

Each stage of the general communication scheme represented in Fig. 2.1 can have a particular implementation depending on the kind of transmission which is considered. Therefore, accounting for all these possibilities in this work would have been somewhat unrealistic. Instead, we will focus on the particular case of the synchronization of a bandpass linearly-modulated signal. The model which will be used in the remainder of this work is introduced in the next section.

2.2 Bandpass Linearly-modulated Signal Transmission

As mentioned in the previous subsection, the implementation of the optimal receiver is usually a computationally-intractable task. In fact, this complexity of implementation is partially due to the fact (see Chapter 4) that the digital-processing device has to deal with the random nature of the parameters characterizing the communication channel . The number and the nature of these parameters may vary as a function of the considered communication scheme. In this section, we emphasize some channel parameters which may appear when considering the transmission of a bandpass linearly-modulated signal. The overall communication scheme is represented in Fig. 2.2.

We assume that we have to transmit a sequence of L information bits $\mathbf{u} = (u_1, u_2, ..., u_L)$ delivered by a source coder. The compression device is assumed to be ideal so that u_i's are i.i.d. The channel coding is performed in two steps. First, information sequence $\mathbf{u}$ is coded using a binary coder with code rate R:

$$\mathbf{x} = \chi(\mathbf{u}) \qquad \text{with } \mathbf{x} \in \mathcal{C}. \tag{2.1}$$

No restriction is imposed on the kind of binary code $\mathcal{C}$ which is used. The

resulting bit sequence $\mathbf{x}$ is then mapped onto a unit-energy signaling constellation $\mathcal{A}$ of size M i.e.

$$\mathbf{a} = \varphi(\mathbf{x}) \qquad \text{with } \mathbf{a} \in \mathcal{A}^K, \tag{2.2}$$

where K is the number of symbols in the transmitted burst. Again, no restriction is imposed on the function $\varphi(\mathbf{x})$ which maps the sequence of coded bits $\mathbf{x}$ onto the sequence of symbols $\mathbf{a}$.

The symbol sequence is shaped by a unit-energy square-root raised-cosine pulse $p(t)$ with roll-off ζ, leading to the following baseband signal

$$s(t) = \sum_{k=0}^{K-1} a_k \, p\,(t - kT)\,, \tag{2.3}$$

where T is the symbol duration. The baseband signal $s(t)$ is then modulated onto a sinusoid with frequency f_T and phase θ_T, such that the resulting waveform fits the frequency range available for the transmission, i.e.

$$s_{PB}(t) = \mathrm{Re}\big\{ s(t)\, e^{j(2\pi f_T\, t + \theta_T)} \big\}. \tag{2.4}$$

The passband signal is transmitted through the physical channel. In this work, we will assume that the channel has the following impulse response,

$$c(t) = A\, \delta(t - \tau), \tag{2.5}$$

i.e. the channel modifies the amplitude and delays the transmitted signal by a factor A and τ respectively. We assume moreover that the physical channel corrupts the transmitted signal by an additive white Gaussian noise (AWGN) $w_{BP}(t)$ with two-sided power spectral density equal to $N_0/2$. The received bandpass signal may therefore be written as

$$\begin{aligned} r_{PB}(t) &= s_{PB}(t) \otimes c(t) + w_{BP}(t), \\ &= A\; s_{PB}(t - \tau) + w_{BP}(t). \end{aligned} \tag{2.6}$$

where $\otimes$ denotes the convolution operator. At the receiver, the bandpass signal is converted to a baseband signal. This is done by multiplying the received signal by a local carrier with frequency f_R and phase θ_R and a low-pass filtering operation. It results in

$$r(t) = \; A \sum_{k=0}^{K-1} a_k \, p(t - kT - \tau)\, e^{j(2\pi \nu t + \theta)} + w(t), \tag{2.7}$$

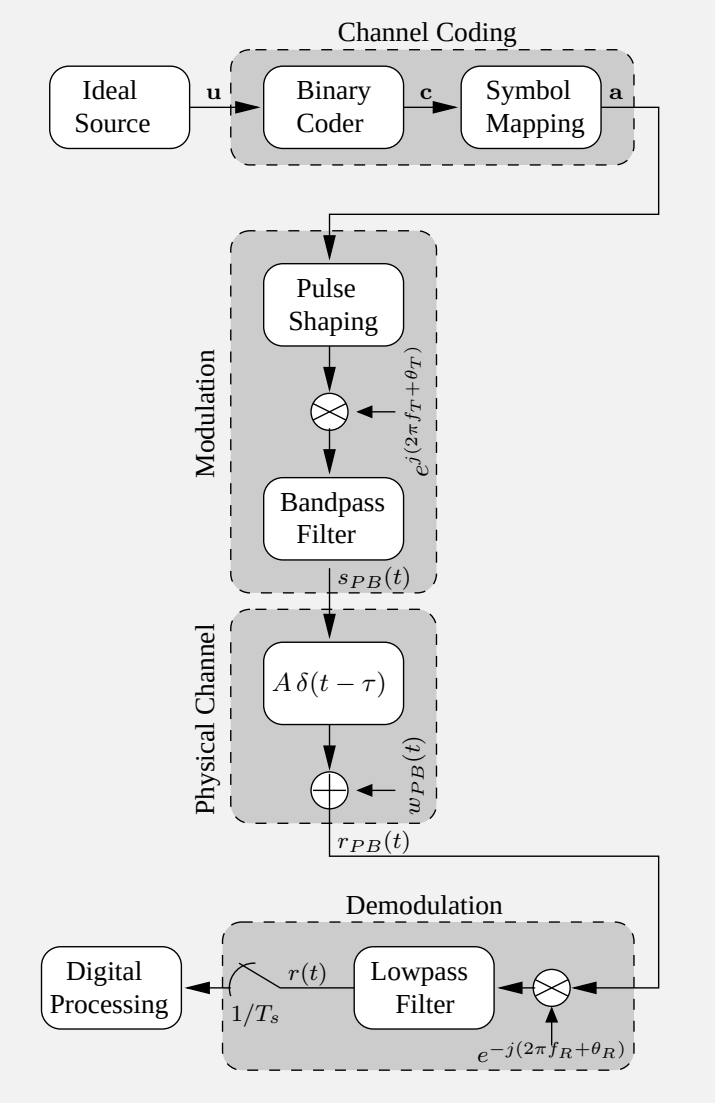

Figure 2.2 Communication scheme for the transmission of a bandpass linearly-modulated signal.

where $w(t)$ is the complex envelope of $w_{BP}(t)$ whereas $\nu = f_T - f_R$ and $\theta = \theta_T - \theta_R$ denote respectively the frequency and the phase mismatch between the local oscillators at the transmitter and receiver sides. ν is usually referred to as *carrier frequency offset* and θ as *carrier phase offset*.

The baseband analog signal $r(t)$ is finally sampled at a rate of $1/T_s$ (with $1/T_s > 2\nu + (1+\zeta)/T$) leading to samples

$$r(lT_s) = A \sum_{k=0}^{K-1} a_k \, p(lT_s - kT - \tau) \, e^{j(2\pi\nu lT_s + \theta)} + w(lT_s), \tag{2.8}$$

where $w(lT_s)$ is a Gaussian noise with variance $\sigma_w^2 = \frac{2N_0}{T_s}$. Since the sampling rate satisfies Nyquist's criterion [21], the samples $r(lT_s)$ contains the same information as the continuous-time signal $r(t)$ and may therefore be fed to the digital-processing device to estimate the transmitted data $\mathbf{u}$. In the sequel, we will denote by $\mathbf{r}$ the vector containing all the observation samples $r(lT_s)$.

Observe that the samples $r(lT_s)$ not only depend on the (coded) transmitted data a_k but also on other unknown parameters: the channel gain A, the carrier phase offset θ, the carrier frequency offset ν and the timing offset τ. The carrier phase offset and the timing offset are usually further broken up as

$$\theta = k_\theta \Psi + \epsilon_\theta \qquad \text{with } -\frac{\Psi}{2} \leq \epsilon_\theta \leq \frac{\Psi}{2}, \tag{2.9}$$

$$\tau = k_\tau T + \epsilon_\tau \qquad \text{with } -\frac{T}{2} \leq \epsilon_\tau \leq \frac{T}{2}, \tag{2.10}$$

where k_θ and k_τ are integers whereas Ψ is the smallest angle of rotational symmetry of the constellation[1]. The uncertainty we have about the carrier phase offset θ (resp τ) is therefore broken into the uncertainty we have about ϵ_θ (resp. ϵ_τ) and k_θ (resp. k_τ). All the unknown parameters (but the data) affecting the observations are often referred to as *nuisance parameters* since they are not of interest while adding some uncertainty to the received observations. In the sequel the vector of nuisance parameters will be denoted by $\mathbf{b}$.

The goal of the digital-processing part of the receiver is to compute a proper estimate $\hat{\mathbf{u}}$ of the transmitted data by taking into account that the received observations also depend on a nuisance-parameter vector $\mathbf{b}$. This problem is tackled in the remainder of this work.

[1] For M-PSK constellation for example $\Psi = 2\pi/M$ whereas for M-QAM constellation we have $\Psi = 2\pi/4$.

Estimation Theory

3

Estimation theory may be seen as the problem of estimating the values of a random process from some observations of a related random process. This field of science has actually a very rich history. Its formative stages can be attributed to illustrious investigators such as Laplace, Gauss, Legendre, and others. The most important advances in the field occurred however in the 20^{th} century with the work of Wiener, Kolmogorov,... Since then, estimation theory has encountered applications in many areas of sciences and engineering, e.g., communications, control, signal processing, geophysics, econometrics and statistics.

In this chapter we recall the fundamental principles of estimation theory. In particular, the main estimators are reviewed and their main characteristics emphasized. We then introduce the concept of *sufficient statistics* or in other words, we answer the question "When is a data set equivalent to another one regarding the estimation of a parameter?" Finally, the question of the ultimate accuracy of any estimator is tackled. In particular, we present the main expressions of the well-known Cramer-Rao Lower Bound [33], [34].

3.1 Basic Principles

Let Θ denote a parameter vector and $\mathbf{r}$ an observation vector which is a function of Θ. Assume these vectors are related according to some function $f(\cdot)$, i.e.

$$\mathbf{r} = f(\Theta). \tag{3.1}$$

The problem of estimating a vector parameter Θ may simply be formulated as follows: for a given realization of the observations $\mathbf{r}$, how can I make a proper decision, say $\hat{\Theta}$, about the value of Θ? The meaning of "proper decision" may vary according to the nature of the problem we are coping with or the statistical information we have at our disposal.

In fact, any estimation algorithm may be seen as a transformation of a random data vector $\mathbf{r}$, i.e.

$$\hat{\Theta} = g(\mathbf{r}). \tag{3.2}$$

As a consequence, since $\mathbf{r}$ is random $\hat{\Theta}$ is itself random. The efficiency of an estimator is thus usually measured with respect to its statistical properties. In this chapter we present two statistical criteria which will be of interest in the sequel of this dissertation, namely the mean square error and the probability of wrong detection:

- **The Mean Square Error (MSE):** The MSE is a criterion which is often used in the context of *continuous parameter* estimation. Formally the MSE of an estimator may be expressed as

$$\begin{aligned} \text{MSE} &= E_{\Theta,\mathbf{r}}\Big[\big(\hat{\Theta}(\mathbf{r}) - \Theta\big)^{\text{H}}\big(\hat{\Theta}(\mathbf{r}) - \Theta\big)\Big], \\ &= \int_{\mathbf{r}}\int_{\Theta}\Big[\big(\hat{\Theta}(\mathbf{r}) - \Theta\big)^{\text{H}}\big(\hat{\Theta}(\mathbf{r}) - \Theta\big)\Big]\, p(\Theta|\mathbf{r})\, d\Theta\; p(\mathbf{r})\, d\mathbf{r}. \end{aligned} \tag{3.3}$$

  The MSE may actually be seen as a measure of how far, in the average, parameter estimate $\hat{\Theta}$ is from the actual parameter value Θ.

- **The probability of wrong decision:** In the context of *discrete parameter* estimation we are often interested in making a decision $\hat{\Theta}$ about Θ such that the probability of committing an error is low. Formally, the probability of a wrong decision about the value of Θ, say $p(e_\Theta|\hat{\Theta})$, may be expressed as

$$p(e_\Theta|\hat{\Theta}) = \int_{\mathbf{r}}\Big[1 - p(\Theta = \hat{\Theta}|\mathbf{r})\Big]\, p(\mathbf{r})\, d\mathbf{r}. \tag{3.4}$$

The optimal estimators according to criteria (3.3) and (3.4) are respectively implemented by the Minimum Mean Square Error (MMSE) and the Maximum

A Posteriori (MAP) estimators (see sections 3.2 and 3.3). Both estimators require the knowledge of a posteriori distribution $p(\Theta|\mathbf{r})$. Note that probability $p(\Theta|\mathbf{r})$ provides the maximum statistical information we have about parameter Θ. It may be expressed as

$$p(\Theta|\mathbf{r}) \propto p(\mathbf{r}|\Theta)\, p(\Theta), \tag{3.5}$$

where $p(\Theta)$ represents the statistical information about Θ which is available prior to any observation, and $p(\mathbf{r}|\Theta)$ is the statistical information about Θ inherent to the observation of $\mathbf{r}$. In practice, $p(\Theta)$ and $p(\mathbf{r}|\Theta)$ are not always available. Therefore, parameter decision rules based on the knowledge of $p(\Theta|\mathbf{r})$ can not always be derived. Furthermore, the implementation of the optimal MMSE and MAP estimators may exhibit an intractable computational complexity. In such cases, suboptimal alternatives have to be considered. Some of these suboptimal approaches are presented in sections 3.4 and 3.5.

3.2 Minimum-Mean-Square-Error Estimation

Minimum-Mean-Square-Error (MMSE) estimator intends to compute the estimate which minimizes (3.3). From (3.3) it is clear that $\hat{\Theta}_{MMSE}$ has to be such that

$$\int_{\Theta} \left[\left(\hat{\Theta}(\mathbf{r}) - \Theta\right)^{\mathrm{H}} \left(\hat{\Theta}(\mathbf{r}) - \Theta\right) \right] p(\Theta|\mathbf{r})\, d\Theta \quad \text{is minimum } \forall\ \mathbf{r}.$$

The solution of this problem is given [35] by

$$\hat{\Theta}_{MMSE} = E_{\Theta|\mathbf{r}}\left[\Theta\right], \tag{3.6}$$

$$= \int_{\Theta} \Theta\ p(\Theta|\mathbf{r})\, d\Theta, \tag{3.7}$$

i.e. reduces to evaluating the mean of a posteriori distribution $p(\Theta|\mathbf{r})$. In the particular case of Θ and $\mathbf{r}$ jointly Gaussian, the MMSE estimator takes the following simple form:

$$\hat{\Theta}_{MMSE} = \mathbf{m}_{\Theta} + \mathbf{P}_{\Theta\mathbf{r}}\, \mathbf{P}_{\mathbf{rr}}^{-1}\, (\mathbf{r} - \mathbf{m}_{\mathbf{r}}). \tag{3.8}$$

However, in the general case, the MMSE estimator is often quite complex to implement: we must first compute $p(\Theta|\mathbf{r})$ and then perform the requisite integrations to obtain $\hat{\Theta}_{MMSE}$. In section 3.5 we give suboptimal versions of the MMSE estimator which exhibit a more tractable complexity.

3.3 Maximum-A-Posteriori Estimation

As mentioned in section 3.1, another approach of practical interest consists in making the decision about Θ which minimizes the probability of committing an error. From (3.4), we see that this kind of estimator has to be such that

$$p(e_\Theta|\hat{\Theta}) = \int_{\mathbf{r}} \left(1 - p(\Theta = \hat{\Theta}|\mathbf{r})\right) p(\mathbf{r})\, d\mathbf{r} \quad \text{is minimum } \forall\ \mathbf{r}.$$

The solution of this problem is the Maximum-A-Posteriori (MAP) estimator:

$$\hat{\Theta}_{MAP} = \arg\max_{\tilde{\Theta}}\ p(\Theta = \tilde{\Theta}|\mathbf{r}), \tag{3.9}$$

i.e. the value of $\hat{\Theta}$ is found that maximizes $p(\Theta|\mathbf{r})$. Since the logarithm function is monotonically increasing, the MAP estimator may also be rewritten as

$$\hat{\Theta}_{MAP} = \arg\max_{\tilde{\Theta}}\ \log p(\Theta = \tilde{\Theta}|\mathbf{r}). \tag{3.10}$$

Using Bayes rule and noticing that $p(\mathbf{r})$ does not depend on Θ, we get the following alternative expression:

$$\hat{\Theta}_{MAP} = \arg\max_{\tilde{\Theta}} \Big\{ \log p(\mathbf{r}|\Theta = \tilde{\Theta}) + \log p(\Theta = \tilde{\Theta}) \Big\}. \tag{3.11}$$

The MAP estimator is often used in the case of discrete-parameter estimation because, as mentioned above, it minimizes the probability of making a wrong decision. However, MAP estimation also exhibits attractive features for continuous-parameter estimation. In particular, we may notice that if the mean of distribution $p(\Theta|\mathbf{r})$ corresponds to its mode, MAP and MMSE estimation are equivalent. This is the case for example for Gaussian distributions.

3.4 Maximum-Likelihood Estimation

MMSE and MAP estimators are optimal estimation methods exploiting the whole statistical information available about Θ from the observations $\mathbf{r}$, namely $p(\Theta|\mathbf{r})$. In practice however, some information enabling to evaluate $p(\Theta|\mathbf{r})$ may be missing. Thus, suboptimal variants are considered. For example, when parameter a priori information $p(\Theta)$ is not available, a suboptimal variant of the MAP estimator which is often considered is the Maximum-Likelihood (ML) estimator:

$$\hat{\Theta}_{ML} = \arg\max_{\tilde{\Theta}}\ p(\mathbf{r}|\Theta = \tilde{\Theta}). \tag{3.12}$$

Due to the monotonicity of the logarithm function, an alternative version of the ML criterion may be written as

$$\hat{\Theta}_{ML} = \arg\max_{\tilde{\Theta}} \log p(\mathbf{r}|\Theta = \tilde{\Theta}). \tag{3.13}$$

Comparing MAP (3.11) and ML (3.13) expressions, we note that the ML estimator drops a part of the statistical information, namely the parameter a priori information. The two estimators are thus equivalent when $p(\Theta)$ is uniform. In the general case, the MAP and ML estimates are usually asymptotically identical to one another [36]. Indeed, in the large sample case the knowledge of the observations usually swamps that of prior distribution.

In the particular case where the model relating the observations $\mathbf{r}$ to Θ is linear and Gaussian, i.e.

$$\mathbf{r} = \mathbf{H}\Theta + \mathbf{w} \tag{3.14}$$

where $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{P_{ww}})$ and $\mathbf{H}$ is a deterministic matrix, the ML maximization problem (3.13) has the simple following solution:

$$\hat{\Theta}_{ML} = \left(\mathbf{H}^H \mathbf{P}_{\mathbf{ww}}^{-1} \mathbf{H}\right)^{-1} \mathbf{H}^H \mathbf{P}_{\mathbf{ww}}^{-1} \mathbf{r}. \tag{3.15}$$

In the general case, maximizing $p(\mathbf{r}|\Theta)$ is usually quite complex and requires mathematical programming. Nevertheless, ML estimation often remains an attractive choice in practice because of its very good large-sample properties: it is consistent, asymptotically efficient and asymptotically Gaussian with mean

$$E_{\mathbf{r}|\Theta}[\hat{\Theta}_{ML}(\mathbf{r})] = \Theta, \tag{3.16}$$

and covariance matrix

$$E_{\mathbf{r}|\Theta}\left[\left(\hat{\Theta}_{ML}(\mathbf{r}) - \Theta\right)\left(\hat{\Theta}_{ML}(\mathbf{r}) - \Theta\right)^H\right] = \mathbf{F}^{-1}, \tag{3.17}$$

where $\mathbf{F}$ is the Fisher information matrix (see section 3.7).

3.5 Estimators Constrained to a Particular Mathematical Structure

Although leading to very good estimation quality, MMSE, MAP or ML estimators are often quite complex to implement. An approach to cope with

this problem consists in constraining the estimator to a particular mathematical structure. In this section we will distinguish between the Linear MMSE (LMMSE) estimator and the Best Linear Unbiased (BLU) estimator.

3.5.1 Linear MMSE Estimator

The estimator is constrained to be an affine transformation of $\mathbf{r}$, i.e.

$$\hat{\Theta} = \mathbf{M}\,\mathbf{r} + \mathbf{q},$$

where $\mathbf{M}$ and $\mathbf{q}$ are so that

$$\begin{aligned} &E_{\Theta,\mathbf{r}}\left[\hat{\Theta}\right] = \mathbf{m}_{\Theta}, \\ &E_{\Theta,\mathbf{r}}\left[(\hat{\Theta}-\Theta)^H(\hat{\Theta}-\Theta)\right] \quad \text{is minimum.} \end{aligned}$$

In other words, we still want the estimator to minimize the MSE but under some additional constraints. The solution of this constrained minimization problem is given [35] by

$$\hat{\Theta}_{LMMSE} = \mathbf{m}_{\Theta} + \mathbf{P}_{\Theta\mathbf{r}}\,\mathbf{P}_{\mathbf{rr}}^{-1}\,(\mathbf{r}-\mathbf{m}_{\mathbf{r}}). \tag{3.18}$$

Comparing (3.8) and (3.18), we see that the MMSE and LMMSE estimators are equivalent if Θ and $\mathbf{r}$ are jointly Gaussian. In the particular case of a Gaussian linear model, i.e. $\mathbf{r} = \mathbf{H}\Theta + \mathbf{w}$, (3.18) reduces [35] to

$$\hat{\Theta}_{LMMSE} = \mathbf{m}_{\Theta} + \mathbf{P}_{LMMSE}\,\mathbf{H}^H\mathbf{P}_{\mathbf{ww}}^{-1}\,(\mathbf{r}-\mathbf{m}_{\mathbf{r}}), \tag{3.19}$$

where

$$\mathbf{P}_{LMMSE} = \left(\mathbf{P}_{\Theta\Theta}^{-1} + \mathbf{H}^H\mathbf{P}_{\mathbf{ww}}^{-1}\mathbf{H}\right)^{-1}. \tag{3.20}$$

3.5.2 Best Linear Unbiased Estimator

The Best Linear Unbiased (BLU) estimator is a conditionally unbiased and minimum-variance estimator which is constrained to a linear structure, i.e.

$$\hat{\Theta} = \mathbf{M}\,\mathbf{r}, \tag{3.21}$$

with $\mathbf{M}$ such that

$$E_{\mathbf{r}|\Theta}[\hat{\Theta}] = \Theta, \tag{3.22}$$

$$E_{\mathbf{r}|\Theta}\left[(\hat{\Theta}-\Theta)^H(\hat{\Theta}-\Theta)\right] \quad \text{is minimum.} \tag{3.23}$$

This constrained minimization problem does often not have any solution since objective function $E_{\mathbf{r}|\Theta}\left[(\hat{\Theta}-\Theta)^H(\hat{\Theta}-\Theta)\right]$ explicitly depends on the parameter to estimate, i.e. Θ. In the particular case $\mathbf{r}=\mathbf{H}\Theta+\mathbf{w}$, a solution exists and may be written as

$$\hat{\Theta}_{BLU}=\left(\mathbf{H}^H\,\mathbf{P}_{\mathbf{ww}}^{-1}\,\mathbf{H}\right)^{-1}\mathbf{P}_{\mathbf{ww}}^{-1}\,\mathbf{H}\,\mathbf{r}. \tag{3.24}$$

Comparing (3.19) and (3.24), we see that $\hat{\Theta}_{LMMSE}$ is equivalent to $\hat{\Theta}_{BLU}$ when $\mathbf{P}_{\Theta\Theta}^{-1}\rightarrow 0$. In the same way, $\hat{\Theta}_{BLU}$ corresponds to the ML solution (3.15) when the noise which affects the observations is Gaussian.

3.6 Sufficient Statistics

In this section, we introduce the concept of sufficient statistics. Sufficient conditions for *sufficiency* are given through the *factorization theorem*. We apply the latter theorem to show that the matched-filter outputs are sufficient statistics for conditional data estimation.

The observations from which Θ should be estimated, are often received in the form of either a continuous-time signal or a discrete-time signal with a large number of samples. In this context the question may arise of the existence of an equivalent data set, of smaller dimension, such that the estimation of Θ from that reduced data set is as good as using the original data set. Such an equivalent data set is referred to as *sufficient statistic*. A formal definition [35] of a sufficient statistic is given by:

Definition 1 (Sufficient Statistic:) *A statistic $T(\mathbf{r})$ is sufficient for the estimation of Θ, if and only if the distribution of $\mathbf{r}$, conditioned on $T(\mathbf{r})=t$, does not involve Θ.*

Deriving sufficient statistics using Definition 1 is often quite difficult. An equivalent definition of sufficiency, much easier to apply, is given by the following theorem:

Theorem 1 (Factorization Theorem:) *A necessary and sufficient condition for $T(\mathbf{r})$ to be sufficient for Θ is that there exists a factorization*

$$p(\mathbf{r}|\boldsymbol{\Theta})=g\big(T(\mathbf{r}),\Theta\big)\,h(\mathbf{r}), \tag{3.25}$$

where the first factor in (3.25) may depend on Θ, but depends on $\mathbf{r}$ only through $T(\mathbf{r})$, whereas the second factor is independent of Θ.

The proof of the theorem can be found in [37] for the continuous case and [38] for the discrete case.

Example 3.1(Matched-filter outputs for conditional data detection) :

As an example of sufficient statistic in the field of digital communication, we mention the set of matched-filter outputs

$$y_k(\nu,\tau) \triangleq T_s \sum_{l=-\infty}^{+\infty} r(lT_s)\, e^{-j2\pi\nu lT_s}\, p(lT_s - kT - \tau), \tag{3.26}$$

$$= \int_{-\infty}^{+\infty} r(t)\, e^{-j2\pi\nu t}\, p(t - kT - \tau)\, dt, \tag{3.27}$$

for symbol detection when synchronization parameters are perfectly known [21]. Indeed, using model (2.8), $p(\mathbf{r}|\mathbf{a},\mathbf{b})$ may be written as

$$p(\mathbf{r}|\mathbf{a},\mathbf{b}) \sim \exp\left(-\frac{T_s \sum_l \parallel r(lT_s) - A\, a_k\, e^{j(\theta+2\pi\nu lT_s)}\, p(lT_s - kT - \tau) \parallel^2}{2N_0}\right),$$

$$\sim g(\mathbf{y},\mathbf{b})\; h(\mathbf{r}), \tag{3.28}$$

where

$$g(\mathbf{y},\mathbf{b}) = \exp\left(-\frac{\sum_k \parallel A\, a_k \parallel^2}{2N_0}\right) \times \exp\left(\frac{2\sum_k \mathrm{Re}\{A\, a_k^*\, e^{-j\theta}\, y_k(\nu,\tau)\}}{2N_0}\right), \tag{3.29}$$

$$h(\mathbf{r}) = \exp\left(-\frac{T_s \sum_l \|r(lT_s)\|^2}{2N_0}\right). \tag{3.30}$$

From the factorization theorem, we have therefore that the vector of matched-filter outputs $\mathbf{y} = [y_1(\nu,\tau)\ \ldots\ y_K(\nu,\tau)]^T$, with $y_k(\nu,\tau)$ defined in (3.26), represents a sufficient statistic for the estimation of data symbols a_k (as long as the synchronization parameters $\mathbf{b}$ are known). The alternative expression of the matched filter outputs (3.27) derives from the *equivalence theorem* [8]. ■

3.7 Assessment of the Estimator Performance

A natural question which arises when designing estimators is the ultimate accuracy that can be achieved in the estimation operation. In particular, does it exist any estimator enabling to achieve an arbitrary small MSE? The answer to

this question is obviously "No": the MSE is lower-bounded and the minimum MSE is given by

$$\mathrm{MSE}_{\min} = E_{\Theta,\mathbf{r}}\left[\left(\hat{\Theta}_{MMSE}(\mathbf{r}) - \Theta\right)^{\mathrm{H}}\left(\hat{\Theta}_{MMSE}(\mathbf{r}) - \Theta\right)\right], \tag{3.31}$$

i.e. by the MSE achieved by the MMSE estimator (3.6). In many practical situations, the MMSE estimator is far too complex to implement. Hence, suboptimal[1] estimators have to be considered. In such cases, it would be relevant to compare the MSE achieved by these suboptimal approaches to $\mathrm{MSE}_{\min}$ in order to determine whether their performance is close or not to the optimal MMSE solution. The gap between the MSE achieved by a suboptimal estimator and $\mathrm{MSE}_{\min}$ actually gives a measure of the degree of suboptimality of the estimator.

Unfortunately, the computation of the MSE achieved by the optimal MMSE estimator is in general infeasible. In order to circumvent this problem, lower bounds on the achievable MSE have been developed. The Barankin bound for example is a large-error bound, i.e. a bound probing the behavior of the observation over a wide range of parameter values. This bound [39] is the highest lower bound on the achievable estimation mean square error. Its evaluation is however quite complicated and the bound is thus only useful in very seldom cases. The Bhattacharyya bound [40] is a small-error bound, i.e. a bound which draws on the characteristics of the observations near the true parameter value. A particular case of the Bhattacharyya bound is the Cramer-Rao bound [33], [34]. In this dissertation we will consider the family of Cramer-Rao bounds (CRB) [41] which are simpler to evaluate than the other above-mentioned lower-bounds while quite tight in numbers of practical cases. In the remainder of this section, we introduce the Bayesian, the standard and the modified CRBs. For more details, we refer the interested reader to [20] (and references therein) for an interesting discussion about the different kinds of CRBs.

3.7.1 The Bayesian Cramer-Rao Bound

Let $\hat{\Theta}(\mathbf{r})$ be an estimator of Θ such that the integral

$$\int_{\mathbf{r}} \left(\hat{\Theta}(\mathbf{r}) - \Theta\right) p(\mathbf{r}|\Theta)\, d\mathbf{r} \tag{3.32}$$

[1] in terms of the MSE minimization.

can be differentiated under the integral for all the elements Θ_i of Θ. Then, we have that the error covariance matrix of $\hat{\Theta}$ is lower bounded as

$$E_{\Theta,\mathbf{r}}\left[\left(\hat{\Theta}(\mathbf{r})-\Theta\right)\left(\hat{\Theta}(\mathbf{r})-\Theta\right)^{\mathrm{H}}\right] \succeq \mathbf{J}^{-1}, \tag{3.33}$$

where $\mathbf{J}$ is the *Bayesian information matrix* and $\succeq$ denotes that matrix $E_{\Theta,\mathbf{r}}\left[\left(\hat{\Theta}(\mathbf{r})-\Theta\right)\left(\hat{\Theta}(\mathbf{r})-\Theta\right)^{\mathrm{H}}\right] - \mathbf{J}^{-1}$ is positive semi-definite. From (3.33) and using the property that the diagonal elements of a positive semi-definite matrix are greater or equal to zero, we further have that

$$E_{\Theta,\mathbf{r}}\left[\| \hat{\Theta}_i(\mathbf{r}) - \Theta_i \|^2\right] \geq \mathbf{J}_{ii}^{-1}, \tag{3.34}$$

and

$$E_{\Theta,\mathbf{r}}\left[\left(\hat{\Theta}(\mathbf{r})-\Theta\right)^{\mathrm{H}}\left(\hat{\Theta}(\mathbf{r})-\Theta\right)\right] \geq \sum_i \mathbf{J}_{ii}^{-1}. \tag{3.35}$$

Using the elements of the inverse of the Bayesian information matrix $\mathbf{J}$, we may therefore lower-bound both the MSEs relative to the estimation of Θ or any Θ_i. These lower bounds on the MSE are often referred to as the *Bayesian Cramer-Rao bounds*.

The element at row i and column j of the Bayesian information matrix may be equivalently computed via the following expressions:

$$\mathbf{J}_{ij} = E_{\mathbf{r},\Theta}\left[\frac{\partial \log p(\Theta,\mathbf{r})}{\partial \Theta_i}\,\frac{\partial \log p(\Theta,\mathbf{r})}{\partial \Theta_j}\right], \tag{3.36}$$

$$= -E_{\mathbf{r},\Theta}\left[\frac{\partial^2 \log p(\Theta,\mathbf{r})}{\partial \Theta_i\,\partial \Theta_j}\right], \tag{3.37}$$

$$= -E_{\mathbf{r},\Theta}\left[\frac{\partial^2 \log p(\mathbf{r}|\Theta)}{\partial \Theta_i\,\partial \Theta_j}\right] - E_{\Theta}\left[\frac{\partial^2 \log p(\Theta)}{\partial \Theta_i\,\partial \Theta_j}\right], \tag{3.38}$$

$$= E_{\mathbf{r},\Theta}\left[\frac{\partial \log p(\mathbf{r}|\Theta)}{\partial \Theta_i}\,\frac{\partial \log p(\mathbf{r}|\Theta)}{\partial \Theta_j}\right] + E_{\Theta}\left[\frac{\partial \log p(\Theta)}{\partial \Theta_i}\,\frac{\partial \log p(\Theta)}{\partial \Theta_j}\right]. \tag{3.39}$$

See [41] for more details about the connections between relationships (3.36)-(3.39).

3.7.2 The Standard Cramer-Rao Bound

We now consider a particular case which will be of interest in the remainder of this dissertation: we assume that we have no a priori information about Θ,

i.e. $p(\Theta)$ is uniform. Therefore, expression (3.39) reduces to

$$\mathbf{J}_{ij} = E_{\Theta}\big[\,\mathbf{F}_{ij}(\Theta)\,\big], \tag{3.40}$$

where $\mathbf{F}(\Theta)$ is the *Fisher information matrix* whose elements are defined as

$$\mathbf{F}_{ij}(\Theta) = E_{\mathbf{r}|\Theta}\left[\frac{\partial \log p(\mathbf{r}|\Theta)}{\partial \Theta_i}\,\frac{\partial \log p(\mathbf{r}|\Theta)}{\partial \Theta_j}\right], \tag{3.41}$$

$$= -E_{\mathbf{r}|\Theta}\left[\frac{\partial^2 \log p(\mathbf{r}|\Theta)}{\partial \Theta_i \partial \Theta_j}\right]. \tag{3.42}$$

If the Fisher information matrix $\mathbf{F}$ depends on Θ, it means that the observations $\mathbf{r}$ provide more information in the average for some values of Θ. In the applications we will consider in the sequel, this is not the case and the Fisher information matrix is constant for all value of Θ, i.e. $\mathbf{F}(\Theta) = \mathbf{F}$. As a consequence, we have

$$E_{\Theta,\mathbf{r}}\big[\big(\hat{\Theta}(\mathbf{r}) - \Theta\big)\,\big(\hat{\Theta}(\mathbf{r}) - \Theta\big)^{\mathrm{H}}\big] \succeq \mathbf{F}^{-1}. \tag{3.43}$$

Actually it may be shown [35] that inequality (3.43) still holds for the conditional error covariance matrix, i.e.

$$E_{\mathbf{r}|\Theta}\big[\big(\hat{\Theta}(\mathbf{r}) - \Theta\big)\,\big(\hat{\Theta}(\mathbf{r}) - \Theta\big)^{\mathrm{H}}\big] \succeq \mathbf{F}^{-1}. \tag{3.44}$$

The inverse of the Fisher information matrix $\mathbf{F}$ (or at least its elements) is actually a good tool for assessing the estimator performance in a number of practical cases. Indeed, as mentioned at the beginning of this section we should ideally compare the estimator performance with the MSE of the MMSE estimator, say MSE_{MMSE}. By definition of the MMSE estimator, the MSE of any estimator has to be larger or equal to MSE_{MMSE}. In particular,

$$\mathrm{MSE}_{ML} \geq \mathrm{MSE}_{MMSE}. \tag{3.45}$$

If we do not have a priori information about Θ, we also have from (3.43) that

$$\mathrm{MSE}_{MMSE} \geq \sum_i \mathbf{F}_{ii}^{-1}. \tag{3.46}$$

On the other hand, we have that

$$\mathrm{MSE}_{ML} \simeq \sum_i \mathbf{F}_{ii}^{-1}. \tag{3.47}$$

as long as the size of the observation vector is large enough (See ML estimator properties in section (3.4)). Combining (3.45), (3.46) and (3.47) we finally have

$$\mathrm{MSE}_{MMSE} \simeq \sum_i \mathbf{F}_{ii}^{-1}. \tag{3.48}$$

The elements of the inverse of the Fisher information matrix therefore enable to have a tight bound of the best performance in terms of MSE of any estimator[2]. The sum of the diagonal elements of the Fisher information matrix, i.e. $\sum_i \mathbf{F}_{ii}^{-1}$, is often referred to as the *(standard) Cramer-Rao bound*.

3.7.3 The Modified Cramer-Rao Bound

The standard Cramer-Rao bound can be difficult to evaluate when the observations also depend on a random nuisance vector, say Φ. In this case, the likelihood function appearing in (3.41) and (3.42) writes

$$p(\mathbf{r}|\Theta) = E_\Phi[p(\mathbf{r}|\Theta,\Phi)]. \tag{3.49}$$

The evaluation of (3.49) may be quite cumbersome since requiring either a multi-dimensional integral or a summation over a huge number of terms. As a consequence, the Fisher information matrix is difficult to evaluate and so is the CRB. In order to circumvent this problem, a *modified Cramer-Rao bound (MCRB)*, simpler to evaluate than the actual CRB, have proposed in [42]. This lower bound derives from a modified Fisher information matrix $\mathbf{G}$ whose element at row i and column j is given by

$$\mathbf{G}_{ij} = E_\Phi\left[E_{\mathbf{r}|\Theta,\Phi}\left[-\frac{\partial^2 \log p(\mathbf{r}|\Theta,\Phi)}{\partial\Theta_i\partial\Theta_j}\right]\right] \tag{3.50}$$

$$= E_\Phi\left[E_{\mathbf{r}|\Theta,\Phi}\left[\frac{\partial \log p(\mathbf{r}|\Theta,\Phi)}{\partial\Theta_i}\,\frac{\partial \log p(\mathbf{r}|\Theta,\Phi)}{\partial\Theta_j}\right]\right]. \tag{3.51}$$

The modified Cramer-Rao bound associated to the estimation of Θ may then be defined as

$$\mathrm{MCRB} = \sum_i \mathbf{G}_{ii}^{-1}. \tag{3.52}$$

Using the Jensen's inequality for matrices [43], it may be shown that

$$\mathbf{F}^{-1} \succeq \mathbf{G}^{-1}, \tag{3.53}$$

[2] As long as the number of observation is large enough.

which implies

$$E_{\mathbf{r}|\Theta}\left[\left(\hat{\Theta}(\mathbf{r})-\Theta\right)^{\mathrm{H}}\left(\hat{\Theta}(\mathbf{r})-\Theta\right)\right] \geq \mathrm{CRB} \geq \mathrm{MCRB}. \tag{3.54}$$

The MCRB is therefore a looser bound than the CRB. The MCRB has nevertheless been shown in [44] to be a tight lower bound on the MSE as long as the variance of the noise affecting the observations is small. In general, the MCRB is much easier to compute than the true CRB.

3.8 Main Points

In this chapter, we review some of the fundamental notions of estimation theory:

- We first consider (section 3.1) two design criteria which will be of interest in the remainder of this dissertation: the mean square error (MSE) (3.3) and the probability of wrong detection (3.4). The first criterion is usually used in the context of continuous parameter estimation whereas the second is rather of interest for discrete-parameter estimation.
- The estimators which are optimal in terms of the minimization of the MSE and the probability of wrong detection are respectively the Minimum Mean Square Error (MMSE) estimator (section 3.2) and the Maximum A Posteriori (MAP) estimator (section 3.3).
- Maximum-likelihood (ML) estimator (section 3.4) is a particular case of MAP estimation when no a priori information $p(\Theta)$ is available. The ML estimator has very good large-sample properties (and therefore so has the MAP estimator): it is conditionally unbiased, efficient and consistent.
- The Linear MMSE (LMMSE) and the Best Linear Unbiased (BLU) estimators (section 3.5) are two estimators trying to optimize the estimation MSE under some additional constraints. In particular, the LMMSE estimator is constrained to have an affine structure whereas the BLU estimator is constrained to have a linear structure.
- The concept of sufficient statistics is introduced in section 3.6. The Factorization Theorem gives necessary and sufficient conditions for sufficiency. The vector made up with the matched-filter output sampled at symbol-rate is shown to be a sufficient statistic for (conditional) data detection.
- The Cramer-Rao Bound (CRB) is shown (section 3.7) to be a tight lower bound for assessing the MSE achieved by any estimator . The CRB may

be built using the elements of the inverse of the Fisher information matrix. Its evaluation is however sometimes quite cumbersome and a modified Cramer-Rao bound (MCRB) much easier to evaluate has been proposed (see section (3.7.3)). The MCRB is usually looser than the CRB. At high SNR, it may nevertheless be shown that the CRB and the MCRB are equivalent.

Optimal Reception and Synchronization

4

The ultimate goal of any communication system consists in transmitting without errors an information rate equal to the channel capacity. At the transmitter side, this ideal system should therefore shape the data in a proper way to operate at the capacity. This problem is addressed by the *coding theory*. On the other hand, based on the received noisy observations, the receiver should be able to make the best possible decision about the transmitted data. Since the final goal is error-free transmission, the optimal receiver should therefore minimize the probability of wrong decision about the transmitted data. In this chapter, we derive the general expressions of the sequence-wise and bit-wise optimal receivers. We then introduce the concept of *synchronization* and we place it in the context of the optimal reception. It is emphasized that synchronization may be performed under different *synchronization modes*: data-aided (DA), non-data-aided (NDA) and code-aided (CA). We show that only the latter mode is of interest in state-of-the-art receivers and will therefore be the purpose of the remainder of the dissertation.

4.1 Sequence-wise Optimal Reception

In communication protocols, one bit error in the detected sequence $\hat{\mathbf{u}}$ usually leads to the entire rejection of the received packet. As a consequence, one is often interested in minimizing the probability of error on the entire sequence i.e. minimizing the frame error rate (FER). For a given realization of the observation vector $\mathbf{r}$, the probability of making a wrong decision about the transmitted sequence $\mathbf{u}$, say $p(e_{\mathbf{u}}|\mathbf{r})$, writes

$$p(e_{\mathbf{u}}|\mathbf{r}) = 1 - p(\mathbf{u} = \hat{\mathbf{u}}|\mathbf{r}), \tag{4.1}$$

where $\hat{\mathbf{u}}$ is the decision made about $\mathbf{u}$. Considering (4.1), the sequence-wise optimal decision may be written as

$$\hat{\mathbf{u}}_{MAP} = \arg \max_{\tilde{\mathbf{u}} \in \mathcal{U}} \; p(\mathbf{u} = \tilde{\mathbf{u}}|\mathbf{r}). \tag{4.2}$$

The optimal receiver implies therefore to choose, among all the possible sequences, the one which maximizes the a posteriori probability of the transmitted sequence given the received observation vector $\mathbf{r}$. In some cases, the particular factorization of $p(\mathbf{u}|\mathbf{r})$ enables to derive efficient algorithms to search the solution of (4.2) (see chapter 5). The *Viterbi algorithm* [45] for example is an instance of such algorithms in the particular case of a convolutionally-coded transmission.

4.2 Bit-wise Optimal Reception

An alternative choice to sequence-wise optimal reception is bit-wise optimal reception. In the latter case, the decisions made on the transmitted bits have to minimize the probability of committing a bit error. Denoting by e_{u_l} the event of making a wrong decision about transmitted bit u_l, we have

$$p(e_{u_l}|\mathbf{r}) = 1 - p(u_l = \hat{u}_l|\mathbf{r}), \tag{4.3}$$

where $\hat{u}_l$ is the decision made about u_l. From (4.3), the bit-wise optimal decision is the solution of the following maximization problem:

$$\hat{u}_{l,MAP} = \arg \max_{\tilde{u}_l \in \{0,1\}} \; p(u_l = \tilde{u}_l|\mathbf{r}). \tag{4.4}$$

Unlike the sequence-wise optimal case, solving maximization problem (4.4) is straightforward since it only requires two evaluations of the objective function $p(u_l|\mathbf{r})$. This simplification of the maximization problem is however at the expense of an increase of the complexity associated to the computation of marginals $p(u_l|\mathbf{r})$ i.e.

$$p(u_l|\mathbf{r}) = \sum_{\sim\{u_l\}} p(\mathbf{u}|\mathbf{r}). \tag{4.5}$$

In the particular case of convolutionally-coded transmissions, an efficient algorithm to compute such marginals is the *BCJR algorithm* [46]. In the general case however, the computation of probabilities $p(u_l|\mathbf{r})$ usually exhibits an intractable complexity. Recently, tractable suboptimal iterative methods have been proposed for the computation of such marginals, see e.g. [2, 3, 4]. Both

these suboptimal methods and the BCJR algorithm will be derived in the next chapter in the context of the factor-graph representation.

Note that the sequence-wise solution (4.2) is usually different from the bit-wise solution (4.4) i.e. $\hat{\mathbf{u}}_{MAP} \neq [\hat{u}_{1,MAP} \cdots \hat{u}_{L,MAP}]$. In a number of cases, however, both approaches lead almost to the same performance and the bit-wise solution might be preferred if less complex.

4.3 Synchronization: A Suboptimal Approach

In this section, we briefly place *synchronization* in the context of optimal reception. In particular, we highlight the optimal-receiver approximation which leads to the synchronization operations performed in most of the receivers.

Methods such as the Viterbi [45] or the BCJR [46] algorithms, enabling to respectively perform sequence-wise and bit-wise optimal reception, assume that the received observations are only a function of the transmitted bit sequence. In practice, the signal to be processed by the receiver also depends on a variety of other parameters (see chapter 2). In particular, the receiver has to deal with the uncertainty relative to the *synchronization parameters*, e.g. the carrier phase offset θ, the carrier frequency offset ν, the timing delay τ, the channel gain A... Considering this uncertainty, the probability which has to be maximized by the sequence-wise and the bit-wise optimal receivers may respectively be written as

$$p(\mathbf{u}|\mathbf{r}) = \int_{\mathbf{b}} p(\mathbf{b}|\mathbf{r})\, p(\mathbf{u}|\mathbf{r},\mathbf{b})\, d\mathbf{b}, \tag{4.6}$$

$$p(u_l|\mathbf{r}) = \int_{\mathbf{b}} p(\mathbf{b}|\mathbf{r}) \sum_{\mathbf{u}\setminus u_l} p(\mathbf{u}|\mathbf{r},\mathbf{b})\, d\mathbf{b}. \tag{4.7}$$

The first factor under the integral sign, i.e. $p(\mathbf{b}|\mathbf{r})$, represents the information about synchronization parameter $\mathbf{b}$ available from the observations $\mathbf{r}$. The second factors represent the information about either the transmitted sequence or a transmitted bit available from the observation $\mathbf{r}$ for a given assumption on the value of $\mathbf{b}$. Unfortunately, the integration over the synchronization-parameter space in (4.6) and (4.7) does not have any simple analytical solution. Actually, the maximization of (4.6) and (4.7) can be shown to be an NP-hard problem i.e. there does not exist any simple algorithm to

exactly solve them. As a consequence, approximations of the optimal receiver have to be considered in practice.

A common suboptimal approach [8], [9] of the optimal receiver when the observations also depend on some nuisance parameters, consists in approximating $p(\mathbf{b}|\mathbf{r})$ by

$$p(\mathbf{b}|\mathbf{r}) \simeq \delta\big(\mathbf{b} - \hat{\mathbf{b}}\big), \tag{4.8}$$

where $\hat{\mathbf{b}}$ is an estimate of the actual parameter value $\mathbf{b}^\star$. This approximation is motivated by the fact that most state-of-the-art communication systems actually require *by design* an accurate knowledge of the synchronization parameters to work properly. For example, powerful channel codes such turbo codes [22] or LDPC codes [10] are designed under the assumption that the communication channel is perfectly known. As a consequence, in order to work properly the system has also to be designed such that $p(\mathbf{b}|\mathbf{r})$ has the following form:

$$p(\mathbf{b}|\mathbf{r}) \simeq \delta(\mathbf{b} - \tilde{\mathbf{b}}), \tag{4.9}$$

where $\tilde{\mathbf{b}}$ denotes a point in a "sufficiently-close" neighborhood of actual parameter value $\mathbf{b}^\star$. Expression (4.9) means that most of the density of $p(\mathbf{b}|\mathbf{r})$ should be concentrated around a point close to $\mathbf{b}^\star$. In this context, approximation (4.8) makes sense.

Plugging approximation (4.8) into (4.6) and (4.7), we get

$$p(\mathbf{u}|\mathbf{r}) \simeq p(\mathbf{u}|\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}), \tag{4.10}$$

$$\begin{aligned} p(u_l|\mathbf{r}) &\simeq \sum_{\mathbf{u}\setminus u_l} p(\mathbf{u}|\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}), \\ &= p(u_l|\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}), \end{aligned} \tag{4.11}$$

i.e. MAP estimation problems (4.2), (4.4) become *conditional* MAP (CMAP) estimation problems:

$$\hat{\mathbf{u}}_{CMAP} = \arg \max_{\tilde{\mathbf{u}} \in \mathcal{U}} \; p(\mathbf{u} = \tilde{\mathbf{u}}|\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}), \tag{4.12}$$

$$\hat{u}_{l,CMAP} = \arg \max_{\tilde{u}_l \in \{0,1\}} \; p(u_l = \tilde{u}_l|\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}). \tag{4.13}$$

Using approximation (4.8), the receiver may be broken up into two main parts (see Fig. 4.1). In a first part, the receiver has to compute a proper estimate $\hat{\mathbf{b}}$

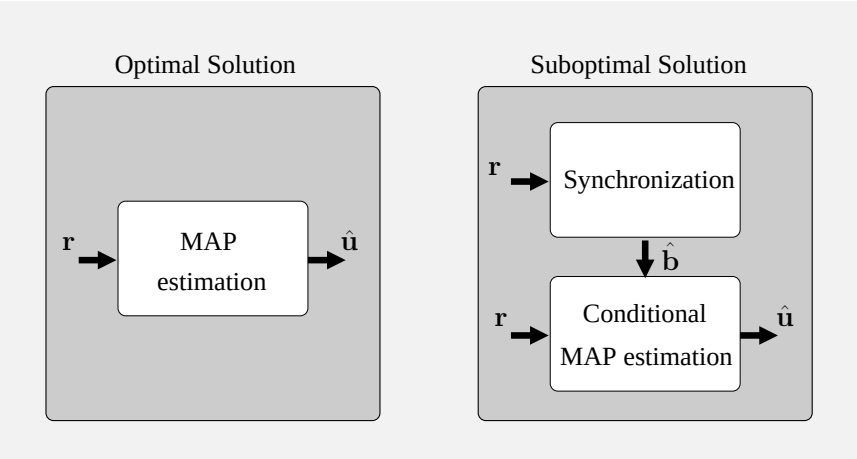

Figure 4.1 Optimal reception versus suboptimal approach based on synchronization.

of $\mathbf{b}$. This task is often referred to as *synchronization*. In a second part, the receiver uses the estimate computed by the synchronization device to estimate the transmitted symbols. Note that using this approach, the overall receiver complexity can remain very close to the complexity of a receiver which would perfectly know synchronization parameter values: the only extra complexity required by this approach is the computation of a proper estimate $\hat{\mathbf{b}}$. The complexity of the synchronization part should therefore remain small as compared to the complexity of the symbol-detection part.

Synchronization is a fundamental operation in digital receivers: its failure usually has catastrophic effects on the system performance[1]. In this dissertation we will therefore focus more particularly on the design of efficient synchronization algorithms.

4.4 Synchronization: Principle

Synchronization algorithms are characterized by a decision rule g which makes a value $\hat{\mathbf{b}}$ correspond to any observation vector $\mathbf{r}$, i.e.

$$\hat{\mathbf{b}} = g(\mathbf{r}). \tag{4.14}$$

[1]This is a consequence of the fact that the channel code assumes that the channel is perfectly known at the receiver side.

In other words, synchronization may actually be viewed as the task of making a good decision about the nuisance-parameter value from an observation vector $\mathbf{r}$ by possibly taking into account some statistical information we have at our disposal.

Considering the model of the received observations in (2.8), we note that samples $r(kT_s)$ contain three different sources of randomness: the additive noise $\mathbf{w}$, the synchronization parameters $\mathbf{b}$ and the transmitted sequence $\mathbf{a}$. The statistical knowledge relative to these random vectors is characterized by the following probabilities: $p(\mathbf{r}|\mathbf{a},\mathbf{b})$, $p(\mathbf{b})$ and $p(\mathbf{a})$. Probability $p(\mathbf{r}|\mathbf{a},\mathbf{b})$ accounts for the distribution of the noise affecting the observations and is typically a Gaussian distribution. Probabilities $p(\mathbf{b})$ and $p(\mathbf{a})$ respectively represent the available a priori information about $\mathbf{b}$ and $\mathbf{a}$. The goal of the synchronizer design consists therefore in taking benefit from this statistical knowledge to built synchronization algorithms whose performance fits some requirements. For example, $p(\mathbf{r}|\mathbf{a},\mathbf{b})$, $p(\mathbf{b})$ and $p(\mathbf{a})$ contains all the necessary information to implement MMSE or MAP estimators (see chapter 3) since $p(\mathbf{b}|\mathbf{r})$ may be computed as

$$p(\mathbf{b}|\mathbf{r}) \propto \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{a})\, p(\mathbf{b}). \tag{4.15}$$

Different design criteria will lead to different synchronizer features. In the sequel, we will consider different approaches to design synchronization algorithms.

4.5 Synchronization Modes

Whatever the design criterion is, the performance of a given synchronization algorithm will depend on the level of randomness which is contained in the observation. Low signal-to-noise ratio for example, will lead to a larger uncertainty in the received observations and the performance of synchronization algorithms will consequently degrade. In the same way, poor a priori information $p(\mathbf{a})$ increases the amount of uncertainty in the received observations and usually leads to worse synchronization performance.

The amount of information about the transmitted sequence provided by the knowledge of a priori probability $p(\mathbf{a})$ depends on the kind of transmission we are considering. The best case would of course consist in the trans-

mission of a *perfectly-known* sequence, say $\mathbf{a}^\star$, so that

$$p(\mathbf{a}) = \delta(\mathbf{a} - \mathbf{a}^\star), \tag{4.16}$$

i.e. there is only one sequence $\mathbf{a}^\star$ for which $p(\mathbf{a})$ does not cancel out. On the contrary, the worst case would consist in a transmission in which all the possible sequences are equiprobable, i.e.

$$p(\mathbf{a}) = \frac{1}{|\mathcal{A}^K|} \qquad \forall\ \mathbf{a} \in \mathcal{A}^K. \tag{4.17}$$

In such a case, the knowledge of $p(\mathbf{a})$ does not bring any information about the transmitted sequence. This case actually corresponds to the transmission of an *uncoded* sequence. Apart from these two limit situations, we can consider an intermediate case for which

$$\begin{aligned} p(\mathbf{a} = \tilde{\mathbf{a}}) &= \frac{1}{|\mathcal{S}|} \qquad \forall\ \tilde{\mathbf{a}} \in \mathcal{S}, \\ &= 0 \qquad \forall\ \tilde{\mathbf{a}} \notin \mathcal{S}. \end{aligned} \tag{4.18}$$

where $\mathcal{S}$ is a subset of $\mathcal{A}^K$. That kind of a priori knowledge corresponds to the case of a *coded* transmission. Indeed, in that kind of scenario, there is only a subset $\varphi(\mathcal{C}) \subset \mathcal{A}^K$ of all possible sequences which corresponds to the legitimate output sequences.

Note that in practice, only scenario (4.18) is of interest in state-of-the-art communication systems. Indeed, on the one hand scenario (4.16) is never considered since the transmission of a known sequence does not carry any information. On the other hand, reliable uncoded transmissions require very high signal-to-noise ratios, which is in contradiction with the requirements of state-of-the-art communication systems. In practice, we usually face coded transmissions and a priori distribution $p(\mathbf{a})$ is therefore distributed according to (4.18).

However, scenari (4.16) and (4.17) are still of interest for the design of synchronization algorithms. Indeed, we may impose the synchronizer to assume that the transmitted sequences are distributed according to (4.16) and (4.17), although the actual sequence a priori distribution is given by (4.18). This kind of approximation usually enables to simplify the derivation of synchronization algorithms. We distinguish between the following *synchronization modes*:

- **Data-aided (DA) mode:** The DA synchronization mode assumes that the transmitted sequences are distributed according to (4.16). This mode

is actually based on the transmission of a sequence of known symbols (pilots), say $\mathbf{a_{pil}}$, among the data symbols. The synchronizer then bases its estimation on the observations samples, say $\mathbf{r_{pil}}$, which only depend on the pilots. The observations corresponding to the data symbols are not taken into account in the estimation process: this leads to the lost of an amount of information. However, the advantage of the DA mode consists in enabling an easy derivation of a close-form expression of synchronizers [8, 9]. Nevertheless, as mentioned above, the pilot symbols do not contain any information. The transmission of pilots required by the DA mode leads therefore to a degradation both in terms of spectral and power efficiency.

- **Non-Data-aided (NDA) mode:** NDA methods make the assumption that the transmitted sequences are distributed according to (4.17). In other words, NDA mode assumes that all the possible sequences are a priori equiprobable. This assumption enables in some cases to ease the derivation of synchronizers [8, 9] which, unlike DA synchronizers, take into account the whole observation vector $\mathbf{r}$. This simplification in the derivation of the synchronizers is however at the expense of a lost of information about the sequence a priori distribution (if the sequence is coded). As a consequence, NDA synchronizers usually exhibit worse performance than their CA counterparts, especially at low SNRs.

- **Code-aided (CA) mode:** By code-aided synchronizers, we will refer to synchronizers which take into account the *actual* sequence a priori distribution $p(\mathbf{a})$. As mentioned above, transmitted sequences are usually distributed according to (4.18). Note however that hypothesis (4.16) and (4.17) are only particular cases of (4.18). CA synchronizers are therefore general synchronization algorithms enabling to take into account any kind of a priori distribution $p(\mathbf{a})$.

DA and NDA synchronization modes are usually considered in conventional synchronization algorithms (see [8, 9]) since they usually enable to derive simple close-form expressions of the synchronizers. However, unlike CA synchronization mode, DA and NDA modes neglect some available statistical information. CA synchronization is the purpose of this dissertation. In the next section we will emphasize that CA operating mode enables to cope with situations where DA and NDA synchronizers are unable to properly synchronize the system.

4.6 Why Code-aided Synchronization ?

As mentioned in the previous section, DA and NDA hypothesis lead to the lost of some information: DA algorithms drop the observations which depend on the data symbols whereas NDA synchronizers do not take into account the a priori information available about the transmitted sequence. A question which naturally arises in this context is the degradation of the estimation quality due to these approximations. In the remainder of this section, we emphasize the degradations which may occur when operating in DA or NDA modes. We consider the case of continuous-parameter estimation and discrete-parameter estimation.

4.6.1 Continuous-parameter Estimation

Performance of continuous-parameter estimators is usually compared in terms of their mean square error (MSE). In this context, the Cramer-Rao bound (CRB) is usually a good measure of the best MSE achievable by any estimator (see chapter 3).

For the sake of illustration, we have represented in Fig. 4.2 the CRB relative to carrier-phase-offset estimation in different synchronization modes. All the CRBs are represented for a frame of 100 QPSK symbols. The DA CRB is also represented for a frame made up of 5 QPSK symbols, which corresponds to the typical number of pilots considered by a DA synchronizer. Convolutional codes with rate $1/2$ and $1/5$, and constraint length equal to 3 are considered. We may notice that

- Considering the same frame length, the DA mode always outperforms CA and NDA modes. Comparing however the DA CRB for a frame of 5 QPSK symbols, we see that a lot of information is lost with respect to the CA and NDA cases.
- The CA CRBs improve when the constraints imposed by the code increase. The CRB relative to a convolutional code with rate $R = 1/5$ is for example always lower than the CRB corresponding to a code with $R = 1/2$.
- At high SNR, all the synchronization modes result to the same performance (when considering the same frame length).

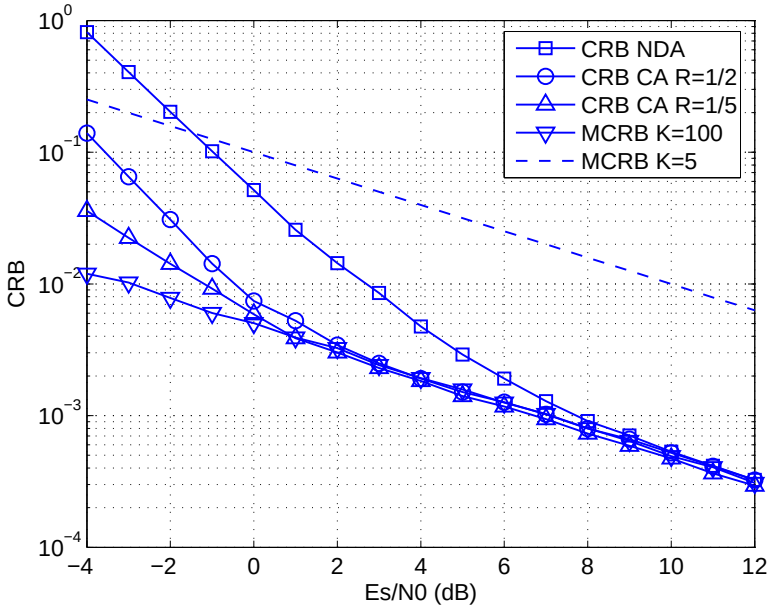

Figure 4.2 Cramer-Rao bounds associated to DA, NDA and CA synchronization modes.

These observations lead to the conclusion that the more a priori information about the transmitted sequence, the better the performance achievable by the synchronizers. However, the limited number of observations considered by practical DA synchronizers degrades heavily the achievable estimation quality. Therefore , this approach is usually not used for the estimation of continuous synchronization parameters.

At high SNR, we note that the amount of a priori information available about the transmitted sequence does no longer affect the estimator performance. This may be understood by the fact that, if the additive noise tends to be equal to zero, the observation of the received signal gives a full knowledge of the transmitted sequence. In this range of SNRs, the NDA approximation will therefore not lead to any degradation with respect to the CA synchronization mode.

At low SNR, we notice that the performance achieved by NDA synchronizers degrades more rapidly than the CA one. Therefore, in communication systems operating at very low SNRs, NDA synchronizers may sometimes be unable to

deliver a sufficient estimation quality to enable a reliable transmission. This problem has particularly been stressed by the discovery of powerful error-correcting codes such as turbo codes [22] or LDPC codes [10]: these codes enable reliable communications at extremely-low SNRs. In receivers operating at such low SNRs, it has been shown that conventional NDA synchronizers operating well at fairly high SNRs are unable to properly synchronize the received signal.

4.6.2 Discrete-parameter Estimation

The performance of discrete-parameter estimators is usually measured in terms of the probability of making a wrong decision about the parameter value. This probability may be written as

$$p(\text{wrong detection}) = \sum_{\tilde{\mathbf{b}} \in \mathcal{B}} p(\hat{\mathbf{b}} \neq \tilde{\mathbf{b}} | \mathbf{b} = \tilde{\mathbf{b}})\, p(\mathbf{b} = \tilde{\mathbf{b}}), \tag{4.19}$$

where $\hat{\mathbf{b}}$ is the decision made about the value of $\mathbf{b}$. In this section, for the sake of comparison between the different synchronization modes, we will consider the minimum error probability which can be achieved by each mode. As mentioned in chapter 3, the probability of wrong detection is minimized by the MAP estimator i.e.

$$\hat{\mathbf{b}}_{MAP} = \arg\max_{\tilde{\mathbf{b}}} p(\mathbf{b} = \tilde{\mathbf{b}} | \mathbf{r}). \tag{4.20}$$

Posterior probability $p(\mathbf{b}|\mathbf{r})$ appearing in (4.20) actually depends on the considered synchronization mode since it may be expressed as in (4.15).

For the sake of example, let us consider the particular case of phase ambiguity resolution for BSPK transmissions. We have therefore $\mathbf{b} = k_\theta$ with $k_\theta \in \{0, 1\}$ and the smallest angle of rotational symmetry of the constellation is $\Psi = \pi$. For the sake of simplicity, we will assume that $p(\mathbf{b})$ is uniform. Using the MAP criterion, a wrong detection on $\mathbf{b} = \tilde{\mathbf{b}}$ occurs when $\hat{\mathbf{b}}_{MAP} \neq \tilde{\mathbf{b}}$ i.e.

$$p(\mathbf{b} = \breve{\mathbf{b}} | \mathbf{r}) > p(\mathbf{b} = \tilde{\mathbf{b}} | \mathbf{r}) \qquad \text{with } \breve{\mathbf{b}} \neq \tilde{\mathbf{b}},$$

or equivalently, since $p(\mathbf{b})$ is assumed uniform, when

$$p(\mathbf{r} | \mathbf{b} = \breve{\mathbf{b}}) > p(\mathbf{r} | \mathbf{b} = \tilde{\mathbf{b}}) \qquad \text{with } \breve{\mathbf{b}} \neq \tilde{\mathbf{b}}. \tag{4.21}$$

Note that $\mathbf{b}$ can only take on two different values and $\Psi = \pi$ so that (4.21) may be rewritten as

$$p(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}}\, e^{j\pi}) > p(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}}). \tag{4.22}$$

Let us consider the probability that (4.22) occurs under the different synchronization modes.

When operating in the DA mode, the likelihood function handled by the synchronizer writes

$$\begin{aligned} p(\mathbf{r}|\mathbf{b}) &= \sum_{\mathbf{a}\in\mathcal{A}^K} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{a}), \\ &= p(\mathbf{r}|\mathbf{a} = \mathbf{a}^\star, \mathbf{b}), \end{aligned} \tag{4.23}$$

where $\mathbf{a}^\star$ is the transmitted sequence. Considering the model of the received observations, we have

$$p(\mathbf{r}|\mathbf{a} = \mathbf{a}^\star, \mathbf{b} = \tilde{\mathbf{b}}\, e^{j\pi}) = p(\mathbf{r}|\mathbf{a} = \mathbf{a}^\star\, e^{j\pi}, \mathbf{b} = \tilde{\mathbf{b}}). \tag{4.24}$$

Since the noise affecting the observation is Gaussian, the probability of

$$p(\mathbf{r}|\mathbf{a} = \mathbf{a}^\star\, e^{j\pi}, \mathbf{b} = \tilde{\mathbf{b}}) > p(\mathbf{r}|\mathbf{a} = \mathbf{a}^\star, \mathbf{b} = \tilde{\mathbf{b}}) \tag{4.25}$$

is given [30] by

$$p(\text{wrong detection}) = \text{erfc}\Big(\sqrt{K\, E_s / N_0}\Big), \tag{4.26}$$

where E_s denotes the mean energy per transmitted symbol and K is the number of symbols in $\mathbf{a}$. The probability of wrong detection when the MAP synchronizer operates in DA mode depends therefore both on the number of transmitted pilots and on the power allocated to the latter ones. An improvement of the synchronizer performance implies therefore either a loss in spectral or power efficiency. In state-of-the-art communication systems, which operate at very low signal-to-noise ratios, the waste of resource associated to the transmission of pilots can become unacceptably large and alternative solutions have therefore to be considered.

Let us consider a synchronizer operating in the NDA mode: the synchronizer assumes that the transmitted sequences are a priori distributed according to (4.17). Therefore, the likelihood function handled by the NDA synchronizer

writes

$$p(\mathbf{r}|\mathbf{b}) = \sum_{\mathbf{a}\in\mathcal{A}^K} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{a}),$$
$$= \sum_{\mathbf{a}\in\mathcal{A}^K} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, \frac{1}{|\mathcal{A}^K|}. \tag{4.27}$$

Taking (4.25) into account and using the fact that $\forall\, \tilde{\mathbf{a}} \in \mathcal{A}^K$, the rotated sequence $\tilde{\mathbf{a}}\, e^{j\pi}$ also belongs to the set of possible transmitted sequence $\mathcal{A}^K$, we have

$$p(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}}) = p(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}}\, e^{j\pi}) \qquad \forall\, \mathbf{r}, \tag{4.28}$$

which in turns leads to

$$p(\text{wrong detection}) = \frac{1}{2}. \tag{4.29}$$

The NDA synchronizer is therefore unable to deal with the phase ambiguity of the received signal. This features is due to the fact that the NDA synchronizer assumes that all the possible sequences exist and therefore, rotating a given sequence by a factor π will also lead to an existing sequence. More generally, the NDA synchronization mode does not enable to solve discrete-parameter synchronization issues.

This problem may be circumvented by using CA synchronization. Indeed, a CA synchronizer takes the actual a priori distribution of the sequences into account. Now, the set of possible transmitted sequence $\mathcal{S}$ is only a subspace of $\mathcal{A}^K$. As a consequence, rotating by a factor π a given sequence $\tilde{\mathbf{a}} \in \mathcal{S}$ will not necessary lead to a sequence belonging to $\mathcal{S}$, i.e. to an existing codeword. The codes having this desirable property are actually well-suited to discrete-parameter estimation: their structure may be used to recover some ambiguities in the received signal without requiring the transmission of pilot symbols.

4.7 Main Points

In this chapter, we introduce the concept of *synchronization* and we place it in the context of optimal data reception. In particular, we emphasize that

- The optimal estimator in terms of the minimization of the frame-error rate (FER) (resp. bit-error rate (BER)) is the sequence (resp. information-bit) MAP estimator.

- The implementation of the optimal MAP estimator is often far too complex. Therefore, instead of using this costive solution, practical receivers often operate in two steps. They first compute an estimate of the synchronization parameters. Then, they perform data detection by assuming that the synchronization parameters are equal to the estimate computed at the first step. The task of computing an estimate of the synchronization parameters is referred to as *synchronization*. Conditional MAP estimation is often considered in the data-detection part of the receiver.
- Three synchronization modes may be distinguished according to the assumption which is made on the sequence a priori distribution $p(\mathbf{a})$: Data-Aided (DA), Non-Data-Aided (NDA) and Code-Aided (CA). CA synchronization mode takes the actual sequence a priori information into account. DA and NDA synchronization modes are approximations.
- The synchronization mode affects the ultimate performance achievable by the receiver. DA and NDA synchronizers lead to satisfactory synchronization performance in receiver operating at medium to high SNRs. In state-of-the-art receivers, which operate at low SNRs, DA and NDA synchronization modes can lead to unacceptable performance degradation or waste of resources. In this case, synchronizers based on the CA mode have to considered.

Factor Graphs and Sum-Product Algorithm

5

The problem of computing marginals of a function naturally arises in a number of situations in communication systems. Most obvious instance is the computation of marginal probabilities $p(\mathbf{u}|\mathbf{r})$ and $p(u_k|\mathbf{r})$, required to implement the optimal-receiver solutions (4.2) and (4.4). The design of efficient algorithms for the computation of such marginals has therefore been addressed a lot of efforts during the last 50 years. Recently, a graphical framework associated with a message-passing algorithm has been proposed [6],[7] for efficient marginal computations: the *factor-graph representation* and the *sum-product (SP) algorithm*. Since then, a large number of practical algorithms for a wide variety of detection and estimation problems have been shown to be special cases or obvious approximations of this framework. In this chapter, we introduce the reader to factor-graph representation and its associated sum-product algorithm. General principles are first presented and then shown to justify (up to a point) some existing receiving schemes. In particular, we show that well-known algorithms such as the Viterbi decoder, the BCJR decoder, the turbo decoder and the turbo demodulator previously proposed in the literature are only particular instances of the SP algorithm. Some parts of this chapter are inspired from the excellent seminal papers [6],[7].

5.1 Marginal Computation: Problem Statement

Let us consider a function $g(\cdot) : (\mathcal{V}_1 \times \mathcal{V}_2 \times ... \times \mathcal{V}_n) \rightarrow \mathcal{R}, (v_1, v_2, ..., v_n) \rightarrow g(v_1, v_2, ..., v_n)$ where $\mathcal{R}$ is a *commutative semiring* with addition $\oplus$ and multiplication $\otimes$. Then, associated with any function $g(v_1, v_2, ..., v_n)$ are n *marginal*

functions $g_i(v_i)$:

$$g_i(v_i) = \overset{\oplus}{\sum_{v_1}} \cdots \overset{\oplus}{\sum_{v_{i-1}}} \overset{\oplus}{\sum_{v_{i+1}}} \cdots \overset{\oplus}{\sum_{v_n}} g(v_1, v_2, ..., v_n), \tag{5.1}$$

$$= \overset{\oplus}{\sum_{\sim\{v_i\}}} g(v_1, v_2, ..., v_n), \tag{5.2}$$

where notation $\overset{\oplus}{\sum_{\sim\{v_i\}}}$ denotes that summation $\oplus$ is performed over all the variables but v_i. This notation will be used throughout the chapter for both discrete-valued and continuous variables. For the continuous case, the "sum" has to be understood as an integral on $\mathcal{V}_i$.

In practice, the computation of marginals $g_i(v_i)$ is often of interest. Codomain $\mathcal{R}$ can actually be any semiring $(\mathcal{R}, \oplus, \odot)$ and therefore many problems may actually be seen as marginalization problems by selecting the proper semiring. Most obvious examples are marginal-probabilities computation in the "sum-product" semiring $(\mathbb{R}, +, \cdot)$ and maximum search in the "max-product" semiring $(\mathbb{R}, \max, \cdot)$ [7]. The remainder of this chapter introduces an efficient way to solve such marginalization problems, namely the factor-graph representation and the sum-product algorithm.

5.2 The Distributive Law and the Factor Graph Representation

An efficient procedure for marginal computation should reduce as much as possible the number of operations needed for evaluating $g_i(v_i)$'s. In order to do so, the semiring structure of codomain $\mathcal{R}$ and more particularly the *distributive law* which applies on it is of central importance [7]. The distributive law states that for any triple (a, b, c) from $\mathcal{R}$, we have

$$(a \odot b) \oplus (a \odot c) = a \odot (b \oplus c), \tag{5.3}$$

The left side of (5.3) involves three arithmetic operations (one addition and two multiplications), whereas the right side needs only two. Thus the distributive law gives us a "fast algorithm" for computing $(a \odot b) \oplus (a \odot c)$. Extending this simple example to the computation of $g_i(v_i)$'s, we note that the way in which function $g(v_1, v_2, ..., v_n)$ factorizes in $\mathcal{R}$ is crucial for the design of efficient marginalization algorithms.

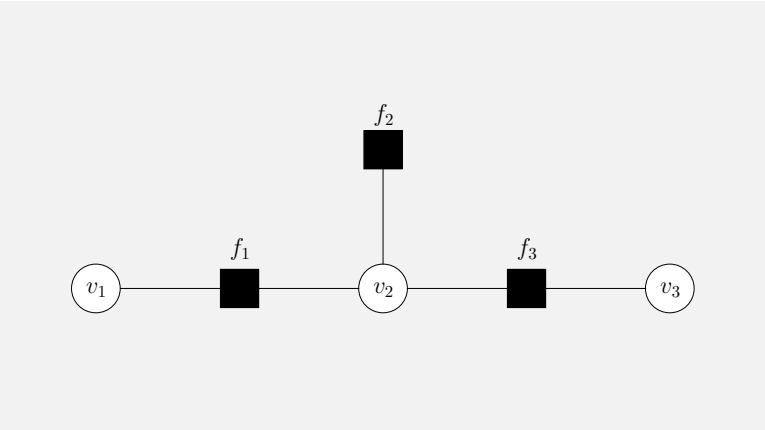

Figure 5.1 Factor graph representation of $g(v_1, v_2, v_3) = f_1(v_1, v_2)\, f_2(v_2)\, f_3(v_2, v_3)$. Factor nodes and variable nodes are respectively denoted by squares and circles.

A *factor graph* is simply a graphical representation of the factorization of a function. Let us assume for example that global function $g(v_1, v_2, ..., v_n)$ can be factorized as

$$g(v_1, v_2, ..., v_n) = \prod_j^{\odot} f_j(v_{Q_j}), \tag{5.4}$$

where v_{Q_j} are sets of elements from $\{v_1, v_2, ..., v_n\}$. The factor graph relative to (5.4) has a variable node for each variable v_i, a factor node for each function f_j and an edge connecting variable node v_i to factor node f_j if and only if v_i is an argument of f_j. The next simple example illustrates the principle of factor-graph representation:

Example 5.1 :

Let $g(v_1, v_2, v_3)$ be a function of three variables and suppose that g may be expressed as a product

$$g(v_1, v_2, v_3) = f_1(v_1, v_2)\, f_2(v_2)\, f_3(v_2, v_3) \tag{5.5}$$

of three factors. The corresponding factor graph in the semiring $(\mathbb{R}, +, \cdot)$ is represented in Fig. 5.1. ■

5.3 The Sum-Product Algorithm

The *sum-product (SP) algorithm* is a message-passing algorithm which operates on factor graphs. Taking benefit from the factor-graph representation, this algorithm enables to compute marginals $g_i(v_i)$ by fully exploiting the factorization of $g(v_1, v_2, ..., v_n)$.

The SP algorithm operates by transmitting messages along the factor-graph edges. Denoting by $\mu_{v_i \to f_j}(v_i)$ the message sent from node v_i to node f_j and by $\mu_{f_j \to v_i}(v_i)$ the message sent from node f_j to node v_i, the message computations performed by the SP algorithm may be expressed as follows:

$$\mu_{v_i \to f_j}(v_i) = \bigodot_{h \in n(v_i) \setminus \{f_j\}} \mu_{h \to v_i}(v_i) \tag{5.6}$$

$$\mu_{f_j \to v_i}(v_i) = \bigoplus_{\sim \{v_i\}} \left(f_j(v_i, v_{Q_j}) \odot \bigodot_{y \in n(f_j) \setminus \{v_i\}} \mu_{y \to f_j}(y) \right) \tag{5.7}$$

where $n(q)$ denotes the set of neighbors of a given node q in the factor graph and v_{Q_j} is a set of elements from $\{v_1, v_2, \cdots, v_n\} \setminus \{v_i\}$. Using these message-update rules on a *cycle-free* factor graph, it can be shown (see Appendix) that marginal $g_i(v_i)$ may be computed as

$$g_i(v_i) = \bigodot_{h \in n(v_i)} \mu_{h \to v_i}(v_i). \tag{5.8}$$

The reasoning made in the appendix is valid for any marginal $g_i(v_i)$. Therefore, applying the SP algorithm on a factor graph without cycles enables to efficiently compute *all* the marginals of the function that the graph represents.

Example 5.2 :

We illustrate the SP algorithm on the simple factor graph introduced in example 5.1. The SP messages can be generated in four steps, as indicated by circles in Fig. 5.2. In details, the messages are generated as follows:

Step 1 :

$$\mu_{v_1 \to f_1}(v_1) = 1$$
$$\mu_{v_3 \to f_3}(v_3) = 1$$
$$\mu_{f_2 \to v_2}(v_2) = \sum_{\sim \{v_2\}} f_2(v_2) = f_2(v_2)$$

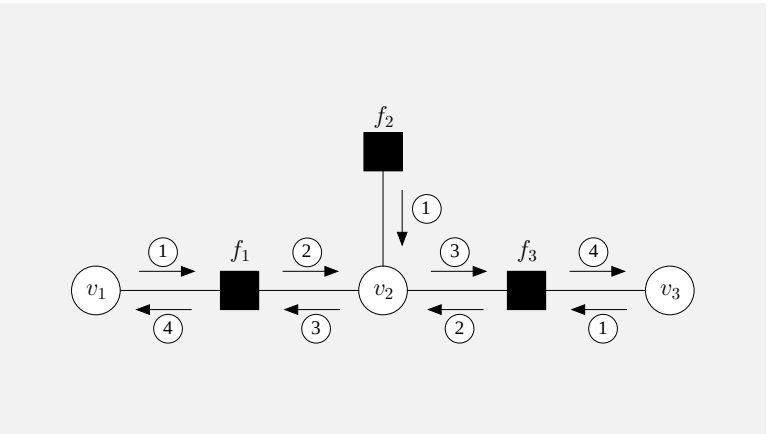

Figure 5.2 Messages generated at each (circled) step of the SP algorithm.

Step 2:

$$\begin{aligned}\mu_{f_1\to v_2}(v_2) &= \sum_{\sim\{v_2\}} f_1(v_1,v_2)\ \mu_{v_1\to f_1}(v_1)\\ &= \sum_{v_1} f_1(v_1,v_2)\\ \mu_{f_3\to v_2}(v_2) &= \sum_{\sim\{v_2\}} f_3(v_2,v_3)\,\mu_{v_3\to f_3}(v_3)\\ &= \sum_{v_3} f_3(v_2,v_3)\end{aligned}$$

Step 3:

$$\begin{aligned}\mu_{v_2\to f_1}(v_2) &= f_2(v_2)\ \mu_{f_3\to v_2}(v_2)\\ &= f_2(v_2)\sum_{v_3} f_3(v_2,v_3)\\ \mu_{v_2\to f_3}(v_2) &= f_2(v_2)\ \mu_{f_1\to v_2}(v_2)\\ &= f_2(v_2)\sum_{v_1} f_1(v_1,v_2)\end{aligned}$$

Step 4:

$$\begin{aligned}\mu_{f_1\to v_1}(v_1) &= \sum_{\sim\{v_1\}} f_1(v_1,v_2)\,\mu_{v_2\to f_1}(v_2)\\ \mu_{f_3\to v_3}(v_3) &= \sum_{\sim\{v_3\}} f_3(v_2,v_3)\,\mu_{v_2\to f_3}(v_2)\end{aligned}$$

The computation of the marginals may then simply be performed accord-

ing to (5.8), i.e.

$$\begin{aligned} g_1(v_1) &= \mu_{f_1 \to v_1}(v_1) \\ &= \sum_{v_2} f_1(v_1, v_2)\, f_2(v_2) \sum_{v_3} f_3(v_2, v_3) \\ g_2(v_2) &= f_2(v_2)\ \mu_{f_1 \to v_2}(v_2)\ \mu_{f_3 \to v_2}(v_2) \\ &= f_2(v_2) \sum_{v_1} f_1(v_1, v_2) \sum_{v_3} f_3(v_2, v_3) \\ g_3(v_3) &= \mu_{f_3 \to v_3}(v_3) \\ &= \sum_{v_2} f_3(v_2, v_3)\, f_2(v_2) \sum_{v_1} f_1(v_1, v_2) \end{aligned}$$

We may notice that the SP algorithm enables to re-use intermediate partial sums which arise in the computation of different marginals. For example, $\sum_{v_3} f_3(v_2, v_3)$ required in $g_1(v_1)$ and $g_2(v_2)$ is only computed once through message $\mu_{f_3 \to v_2}$. ■

The reasoning made in the appendix is valid for factor graphs without cycles. The SP algorithm can however be applied to factor graphs with cycles by using the same message propagation rules (5.6)-(5.7). Because of the cycles in the graph, an iterative algorithm with no natural termination will result, with messages passed multiple times on a given edge. In contrast to the case of no cycles, the result of the SP algorithm can no longer be interpreted as being exact marginals $g_i(v_i)$. The result delivered by the SP algorithm becomes nevertheless all the more accurate as the size of the cycles increases.

5.4 Application of the SP Algorithm to Receiver Tasks

It turns out that many communication problems can be cast in the framework of factor graphs. In fact, a large variety of powerful algorithms in communication theory may be viewed as specific instances of the sum-product algorithm. In this section we replace well-known algorithms such as the Viterbi algorithm [45], the BCJR algorithm [46] and the iterative turbo principle [2], [3] into the context of factor graphs and the SP algorithm.

5.4.1 MAP Sequence Detection

As mentioned in Chapter 4, conditional maximum a posteriori sequence estimation, i.e.[1]

$$\hat{\mathbf{u}}_{CMAP} = \arg\max_{\tilde{\mathbf{u}} \in \mathcal{U}} \; p(\mathbf{u} = \tilde{\mathbf{u}} | \mathbf{y}, \mathbf{b} = \hat{\mathbf{b}}) \tag{5.9}$$

is of interest in practical systems. Unfortunately, the computation of (5.9) is often quite complex since requiring $|\mathcal{U}|$ evaluations of the goal function $p(\mathbf{u}|\mathbf{y}, \mathbf{b})$. In some cases however, the computation of the CMAP sequence estimate may be simplified by exploiting the particular factorization of $p(\mathbf{u}|\mathbf{y}, \mathbf{b})$. This kind of algorithms may be understood by placing the CMAP estimation in the context of marginal computation in the "max-product" semiring[2], i.e $(\mathcal{R}, \oplus, \odot) = (\mathbb{R}, \max, \cdot)$.

Let us set $g(u_1, u_2, ..., u_L) = p(\mathbf{u}|\mathbf{y}, \mathbf{b})$. Marginals $g_i(u_i)$ in the "max-product" semiring may be computed as

$$g_l(u_l) = \max_{\sim\{u_l\}} \; p(\mathbf{u}|\mathbf{y}, \mathbf{b}), \tag{5.10}$$

where $\max_{\sim\{u_l\}}$ denotes maximization over all $u_j \neq u_l$. Conditional MAP sequence estimate $\hat{\mathbf{u}}_{CMAP}$ may then be obtained as

$$\hat{\mathbf{u}}_{CMAP} = [\hat{u}_1, \hat{u}_2, ..., \hat{u}_L]^T, \tag{5.11}$$

where

$$\hat{u}_l = \arg\max_{u_l} \; g_l(u_l). \tag{5.12}$$

The computation of marginals (5.10), and therefore of the CMAP sequence estimate, may be efficiently performed by using factor-graph representation and the SP algorithm.

Example 5.3 (The Viterbi algorithm) :

The Viterbi algorithm [45] applies in the case of BPSK convolutionally-coded transmission and may be viewed as a specific instance of the SP algorithm in the "max-product" semiring $(\mathbb{R}, \max, \cdot)$. Let us consider the factorization of $p(\mathbf{u}|\mathbf{y}, \mathbf{b})$ in this particular case. Note that, since we

[1] The vector made up with the matched-filter output sampled at the symbol rate T, say $\mathbf{y}$, is a sufficient statistic for data estimation (see section 3.6). We consider therefore equivalently vector $\mathbf{y}$ instead of observation vector $\mathbf{r}$.

[2] The presentation of the problem is inspired from [19].

consider a BPSK mapping, there is a one-to-one correspondance between the transmitted symbols a_k and the coded bits x_k, i.e., $a_k = 2x_k + 1$. Therefore, for the sake of simplicity, we will drop the dependence of the observations in the a_k's.

Using Bayes rule and assuming that all the information sequences have the same a priori probability, we have

$$p(\mathbf{u}|\mathbf{y}, \mathbf{b}) \propto p(\mathbf{y}|\mathbf{u}, \mathbf{b}), \tag{5.13}$$

where $\propto$ denotes equality up to an irrelevant factor. Denoting then by $\text{out}(\mathbf{u})$ the decoder output when $\mathbf{u}$ is input, we have

$$\begin{aligned} g(u_l) &= \max_{\sim\{u_l\}} \; p(\mathbf{y}|\mathbf{u} = \tilde{\mathbf{u}}, \mathbf{b}), \\ &= \max_{\sim\{u_l\}} \; p(\mathbf{y}|\mathbf{x} = \text{out}(\mathbf{u}), \mathbf{b})\, p(\mathbf{x} = \text{out}(\mathbf{u})|\mathbf{u}), \end{aligned} \tag{5.14}$$

where the last equality derives from the fact that $p(\mathbf{x}|\mathbf{u}) = 1$ when $\mathbf{x}$ corresponds to the legitimated coder output $\text{out}(\mathbf{x})$. Since the channel is memoryless, factor $p(\mathbf{y}|\mathbf{x}, \mathbf{b})$ factorizes as

$$p(\mathbf{y}|\mathbf{x}, \mathbf{b}) = \prod_{l}^{L} \prod_{n}^{N} p(y_l^n | x_l^n, \mathbf{b}), \tag{5.15}$$

where x_l^n is the n^{th} coded bit corresponding to information bit u_l and y_l^n denotes the associated matched-filter output.

Factor $p(\mathbf{x}|\mathbf{u})$ may also be further factorized by taking into account that the output of a convolutional coder is a Markov process: the output only depends on the current input u_l and on the current state t_l of the coder. Denoting by $\mathbb{I}\{\cdot\}$ the indicator function, which is equal to 1 if the statement in the braces is true and 0 otherwise, we have

$$p(\mathbf{x}|\mathbf{u}) = \mathbb{I}\{t_0 = 0\} \prod_{l}^{L} \mathbb{I}\{t_{l+1} \leftarrow (u_l, t_l)\}\; \mathbb{I}\{x_l^1, ..., x_l^N = \text{out}(u_l, t_l)\}, \tag{5.16}$$

where "$\text{out}(u_l, t_l)$" stands for the coder output corresponding to input u_l and state t_l. The meaning of this expression is the following: $\mathbf{x}$ is the codeword corresponding to $\mathbf{u}$ if $\forall\, l$, both the coder outputs $x_l^1, ..., x_l^N$ and the new state t_{l+1} correspond to its current input u_l and state t_l.

The factor graph relative to factorization (5.15) and (5.16) is represented in Fig. 5.3 and Fig. 5.4. We may observe that the factor graph is cycle-free and therefore applying the SP algorithm on it will lead to the computation of the exact marginals. Using notations from Fig. 5.4, marginal $g_l(u_l)$ is equal to message $\delta_l(u_l)$ entering node u_l. Using SP-algorithm update rules, this message may be computed as

$$\delta_l(u_l) = \max_{\sim\{u_l\}} \alpha_l(t_l)\,\beta_{l+1}(t_{l+1})\,\mathbb{I}\{t_{l+1} \leftarrow (u_l, t_l)\} \times \mathbb{I}\{x_l^1, ..., x_l^N = \text{out}(u_l, t_l)\} \prod_n p(y_l^n | x_l^n, \mathbf{b}), \quad (5.17)$$

where

$$\alpha_l(t_l) = \max_{\sim\{t_l\}} \alpha_l(t_{l-1})\,\mathbb{I}\{t_l \leftarrow (u_{l-1}, t_{l-1})\} \times \mathbb{I}\{x_{l-1}^1, ..., x_{l-1}^N = \text{out}(u_{l-1}, t_{l-1})\} \prod_n p(y_{l-1}^n | x_{l-1}^n, \mathbf{b}), \quad (5.18)$$

$$\beta_l(t_l) = \max_{\sim\{t_l\}} \beta_{l+1}(t_{l+1})\,\mathbb{I}\{t_{l+1} \leftarrow (u_l, t_l)\} \times \mathbb{I}\{x_l^1, ..., x_l^N = \text{out}(u_l, t_l)\} \prod_n p(y_l^n | x_l^n, \mathbf{b}). \quad (5.19)$$

Equations (5.17)-(5.19) are equivalent to the sequence maximum-search procedure proposed by Viterbi [45]. Instead of requiring $2^{|\mathcal{U}|}$ evaluations of objective function $p(\mathbf{u}|\mathbf{y}, \mathbf{b})$, the Viterbi algorithm has a complexity order equal to $L\,2^{|\mathcal{T}|}$ where $|\mathcal{T}|$ is the number of states of the convolutional code. ■

5.4.2 MAP Bit Detection

Another problem of interest in practical receivers is conditional MAP bit estimation, i.e.[3]

$$\hat{u}_{l,BER} = \arg \max_{\hat{u}_l \in \{0,1\}} p(u_l = \hat{u}_l | \mathbf{y}, \mathbf{b} = \hat{\mathbf{b}}), \quad (5.20)$$

where

$$p(u_l | \mathbf{y}, \mathbf{b}) = \sum_{\sim\{u_l\}} p(\mathbf{u} | \mathbf{y}, \mathbf{b}). \quad (5.21)$$

[3]The vector made up with the matched-filter output sampled at the symbol rate T, say $\mathbf{y}$, is a sufficient statistic for data estimation (see section 3.6). We consider therefore equivalently vector $\mathbf{y}$ instead of observation vector $\mathbf{r}$.

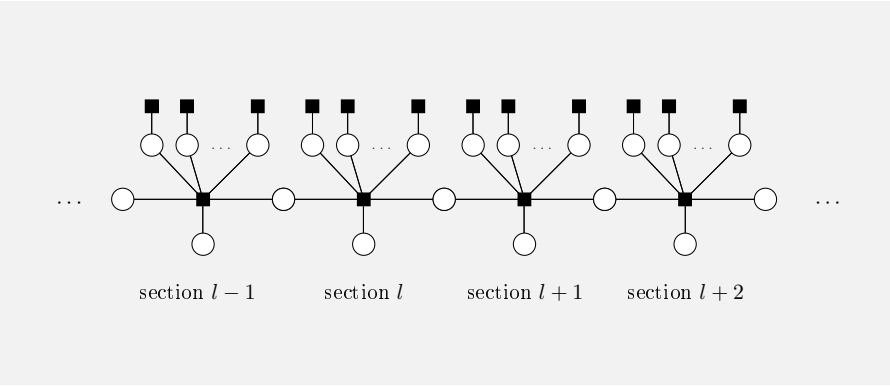

Figure 5.3 Schematic factor-graph representation of $p(\mathbf{y}|\mathbf{x},\mathbf{b})\,p(\mathbf{x}|\mathbf{u})$ for a convolutionally-coded transmission.

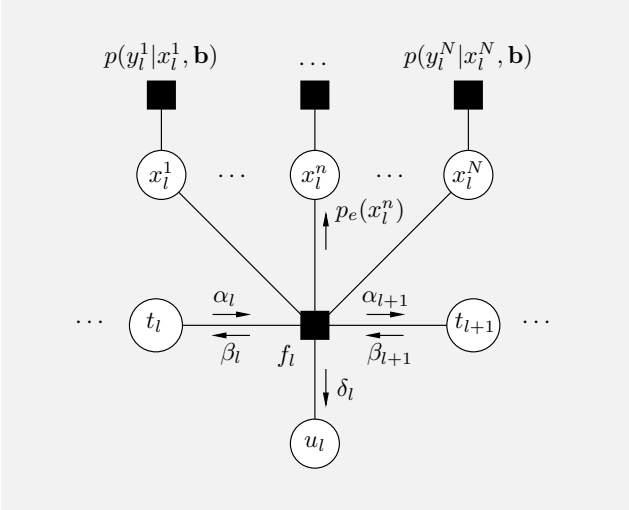

Figure 5.4 Detailed representation of one section of the factor graph of $p(\mathbf{y}|\mathbf{x},\mathbf{b})\,p(\mathbf{x}|\mathbf{u})$ in Fig. 5.3. Node f_l represents factor $f_l = \mathbb{I}\{t_{l+1} \leftarrow (u_l, t_l)\}\ \mathbb{I}\{x_l^1, ..., x_l^N = \text{out}(u_l, t_l)\}$.

Unlike sequence-wise solution (5.9), maximization operation (5.20) is not a problem in this case since it only requires two evaluations of $p(u_l|\mathbf{y},\mathbf{b})$. On the contrary, the computation of marginals $p(u_l|\mathbf{y},\mathbf{b})$ may require quite a large number of summations.

The problem of efficiently computing marginals $p(u_l|\mathbf{y},\mathbf{b})$ may obviously be placed in the context of the SP algorithm in the "sum-product" semiring $(\mathbb{R},+,\cdot)$. Let us consider the particular factorization of $p(\mathbf{u}|\mathbf{y},\mathbf{b})$. Using Bayes rule and assuming that all the information sequences are equiprobable, we have

$$p(\mathbf{u}|\mathbf{y},\mathbf{b}) \propto p(\mathbf{y}|\mathbf{u},\mathbf{b}), \tag{5.22}$$

where $\propto$ denotes equality up to an irrelevant factor. If we assume that the transmission is coded, $p(\mathbf{u}|\mathbf{y},\mathbf{b})$ also writes

$$\begin{aligned} p(\mathbf{u}|\mathbf{y},\mathbf{b}) &\propto \sum_{\mathbf{a},\mathbf{x}} p(\mathbf{y}|\mathbf{a},\mathbf{x},\mathbf{u},\mathbf{b})\, p(\mathbf{a}|\mathbf{x},\mathbf{u})\, p(\mathbf{x}|\mathbf{u}), \\ &\propto \sum_{\mathbf{a},\mathbf{x}} p(\mathbf{y}|\mathbf{a},\mathbf{b})\, p(\mathbf{a}|\mathbf{x})\, p(\mathbf{x}|\mathbf{u}). \end{aligned} \tag{5.23}$$

Marginals probabilities $p(u_l|\mathbf{y},\mathbf{b})$ required for conditional bit-wise MAP estimation may therefore be written as

$$p(u_l|\mathbf{y},\mathbf{b}) \propto \sum_{\sim\{u_l\}} p(\mathbf{y}|\mathbf{a},\mathbf{b})\, p(\mathbf{a}|\mathbf{x})\, p(\mathbf{x}|\mathbf{u}). \tag{5.24}$$

Factor $p(\mathbf{y}|\mathbf{a},\mathbf{b})$ accounts for the effect of the channel whereas $p(\mathbf{a}|\mathbf{x})$ and $p(\mathbf{x}|\mathbf{u})$ respectively model the bit-to-symbol mapping and the code. These probabilities may be further factorized according to the channel and the code which are considered for the transmission. Applying the SP algorithm to the factor graph relative to these factorizations, we get efficient procedures for computing marginals $p(u_l|\mathbf{y},\mathbf{b})$. In the sequel we give the examples of marginal computation for convolutional

Example 5.4 (The BCJR Algorithm) :

In this example, we address the computation of information-bit marginals in the particular case of a BPSK convolutionally-coded transmission through an AWGN channel. In particular, we emphasize that the application of the SP algorithm leads to the well-known BCJR algorithm [46]. We also introduce the notion of *extrinsic probability*. This notion will be shown to be central in the derivation of synchronization algorithms in the remainder of this dissertation.

Since the symbol constellation is BPSK, there is a one-to-one correspondence between coded bits x_l and transmitted symbols a_l. For the sake of conciseness, we can then drop probability $p(\mathbf{a}|\mathbf{x})$ in (5.24). We focus therefore on the following marginalization problem:

$$p(u_l|\mathbf{y},\mathbf{b}) \propto \sum_{\sim\{u_l\}} p(\mathbf{y}|\mathbf{x},\mathbf{b})\, p(\mathbf{x}|\mathbf{u}). \tag{5.25}$$

Note the similarity of marginalization problems (5.14) and (5.25): both consider the marginalization of the same global function, i.e. $p(\mathbf{y}|\mathbf{x},\mathbf{b})\, p(\mathbf{x}|\mathbf{u})$, but in different semiring. In (5.14), marginalization is placed in the "max-product" semiring $(\mathbb{R}, \max, \cdot)$ whereas (5.25) considers the problem in the "sum-product" semiring $(\mathbb{R}, +, \cdot)$. Since the multiplication operation $\odot$ is the same in both semiring, the factor-graph representations relative to these marginalization problems are the same (see Fig (5.3) and Fig. (5.4)). The message-update rules however differ since addition operation $\oplus$ is differently defined in the two semirings.

Let us again observe that the factor graph is cycle-free and therefore applying the SP algorithm on this graph will lead to the computation of the exact marginals. Marginal $p(u_l|\mathbf{y},\mathbf{b})$ is simply equal to the message $\delta_l(u_l)$ entering variable node u_l. Using the SP message update rules, we have

$$\begin{aligned}\delta_l(u_l) = \sum_{\sim\{u_l\}} & \alpha_l(t_l)\, \beta_{l+1}(t_{l+1})\, \mathbb{I}\{t_{l+1} \leftarrow (u_l, t_l)\} \\ & \times \mathbb{I}\{x_l^1, ..., x_l^N = \text{out}(u_l, t_l)\} \prod_n p(y_l^n|x_l^n, \mathbf{b}),\end{aligned} \tag{5.26}$$

where $\alpha_l(t_l)$ and $\beta_l(t_l)$ may be computed as

$$\begin{aligned}\alpha_l(t_l) = \sum_{\sim\{t_l\}} & \alpha_l(t_{l-1})\, \mathbb{I}\{t_l \leftarrow (u_{l-1}, t_{l-1})\} \\ & \times \mathbb{I}\{x_{l-1}^1, ..., x_{l-1}^N = \text{out}(u_{l-1}, t_{l-1})\} \prod_n p(y_{l-1}^n|x_{l-1}^n, \mathbf{b}),\end{aligned} \tag{5.27}$$

$$\begin{aligned}\beta_l(t_l) = \sum_{\sim\{t_l\}} & \beta_{l+1}(t_{l+1})\, \mathbb{I}\{t_{l+1} \leftarrow (u_l, t_l)\} \\ & \times \mathbb{I}\{x_l^1, ..., x_l^N = \text{out}(u_l, t_l)\} \prod_n p(y_l^n|x_l^n, \mathbf{b}).\end{aligned} \tag{5.28}$$

Equations (5.26), (5.27) and (5.28) derived from the factor-graph framework and the application of the SP algorithm exactly correspond to the expressions proposed by Bahl, Cocke, Jelinek and Raviv (BCJR) in [46] for

the computation of marginals $p(u_l|\mathbf{y},\mathbf{b})$ in the case of a convolutionally-coded transmission.

Note that using the same SP update rules, a posteriori probabilities of coded bits may also bit computed as

$$p(x_l^n|\mathbf{y},\mathbf{b}) \propto p(y_l^n|x_l^n,\mathbf{b})\, p_e(x_l^n). \tag{5.29}$$

On the one hand, probability $p(y_l^n|x_l^n,\mathbf{b})$ accounts for the knowledge we have about x_l^n thanks to the observation of y_l^n. On the other hand, $p_e(x_l^n)$ represents the information about x_l^n which is brought, through the code structure, by all the other observations. The latter probability is often referred to as *extrinsic probability* and may be computed as

$$p_e(x_l^n) = \sum_{\sim\{x_l^n\}} \alpha_l(t_l)\,\beta_{l+1}(t_{l+1})\,\mathbb{I}\{t_{l+1} \leftarrow (u_l,t_l)\} \times \mathbb{I}\{x_l^1,...,x_l^N = \text{out}(u_l,t_l)\} \prod_{m\neq n} p(y_l^m|x_l^m,\mathbf{b}). \tag{5.30}$$

As we will see in the sequel, these probabilities will be of central importance in the design of CA synchronization algorithms. ■

Example 5.5 (The Turbo Decoder) :

In this example we show that the famous "turbo principle" introduced for the first time by Berrou [2] for the decoding of turbo code is nothing else than a particular instance of the SP algorithm.

We assume BPSK modulation so that we can consider the equivalent marginalization problem (see Example 5.4):

$$p(u_l|\mathbf{y},\mathbf{b}) \propto \sum_{\sim\{u_l\}} p(\mathbf{y}|\mathbf{x},\mathbf{b})\, p(\mathbf{x}|\mathbf{u}). \tag{5.31}$$

Let us focus on the factorization of $p(\mathbf{y}|\mathbf{x},\mathbf{b})$ and $p(\mathbf{x}|\mathbf{u})$ in the case of a rate-1/3 turbo-coded transmission. The turbo coder consists therefore in two rate-1/2 recursive systematic convolutional (RSC) coders separated by an interleaver. Let us denote y_l as the matched-filter output associated to systematic bit u_l. Let us also define x_l^i as the l^{th} parity bit output by the i^{th} ($i = 1,2$) RSC coder and y_l^i as the corresponding matched-filter output.

Using these definitions, factor $p(\mathbf{y}|\mathbf{x},\mathbf{b})$ can be written as

$$p(\mathbf{y}|\mathbf{x},\mathbf{b}) = \prod_{l}^{L} p(y_l|u_l,\mathbf{b})\, p(y_l^1|x_l^1,\mathbf{b})\, p(y_l^2|x_l^2,\mathbf{b}), \tag{5.32}$$

since the noise affecting the observations is white.

Since the two coders making up the turbo code are convolutional, factor $p(\mathbf{x}|\mathbf{u})$ factorizes in a similar way to (5.16), i.e.

$$\begin{aligned} p(\mathbf{x}|\mathbf{u}) = {} & \mathbb{I}\{t_0^1 = 0\} \prod_{l}^{L} \mathbb{I}\{t_{l+1}^1 \leftarrow (u_l, t_l^1)\}\, \mathbb{I}\{x_l^1 = \mathrm{out}_1(u_l, t_l^1)\} \\ & \times \mathbb{I}\{t_0^2 = 0\} \prod_{l}^{L} \mathbb{I}\{t_{l+1}^2 \leftarrow (u_{\pi(l)}, t_l^2)\}\, \mathbb{I}\{x_l^2 = \mathrm{out}_2(u_{\pi(l)}, t_l^2)\}, \end{aligned} \tag{5.33}$$

where $u_{\pi(l)}$ represents the l^{th} bit of the interleaved sequence.

Combining expressions (5.32) and (5.33), we may build the factor graph represented in Fig. 5.5. Notice that the graph contains cycles. Therefore, the application of the SP algorithm to this graph will not lead to the computation of the exact marginals. The shorter the cycles, the rougher the approximation made by the SP algorithm[4]. The interleaver plays therefore a crucial role since it enables to increase the cycle length in the turbo-decoder graph. Note also that the factor graph in Fig. 5.5 contains the factor graphs of its two constituent convolutional codes. The application of the SP algorithm on these factor graphs should therefore lead to update equations similar to those of the BCJR algorithm.

A consequence of the presence of cycles in the factor graph is that the application of the SP algorithm leads to an iterative algorithm with no natural termination. It is therefore required to define a message-passing schedule which specifies in which order the messages are updated. In the case of a turbo decoder, the messages corresponding to the factor graph of one constituent code are usually first updated. These messages are transmitted through the interleaver and the messages in the factor graph of the second constituent code are in turn updated, etc. As mentioned above, the messages transmitted by the SP algorithm through the edges of the constituent-code factor graphs will therefore be very similar to those of a BCJR algorithm.

[4] Although not proved, this is what we usually observe in practice.

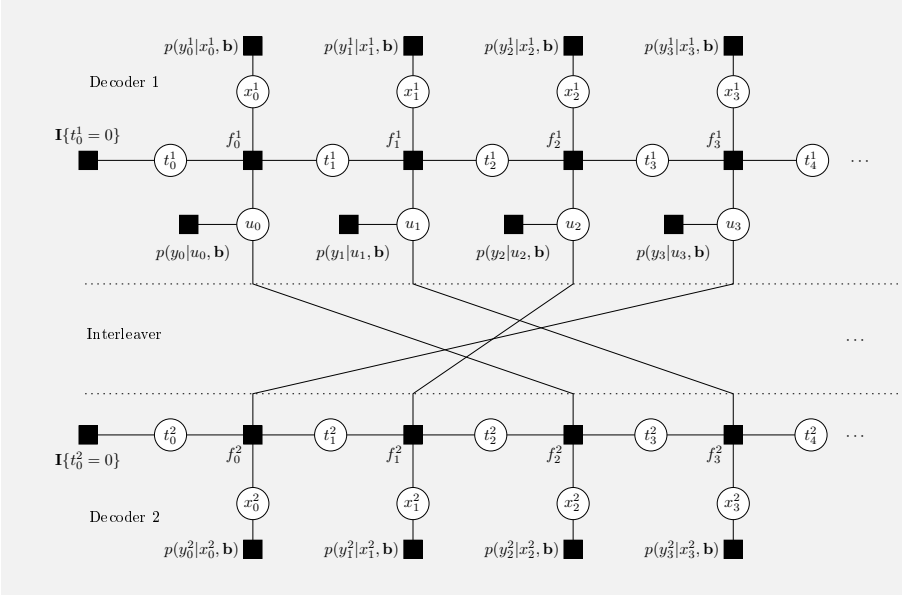

Figure 5.5 Factor-graph representation of $p(\mathbf{y}|\mathbf{x},\mathbf{b})\,p(\mathbf{x}|\mathbf{u})$ for a turbo-coded transmission. Node f_l^i represents $f_l^i = \mathbb{I}\{t_{l+1}^i \leftarrow (u_l, t_l^i)\}\;\mathbb{I}\{x_l^i = \mathrm{out}_i(u_l, t_l^i)\}$.

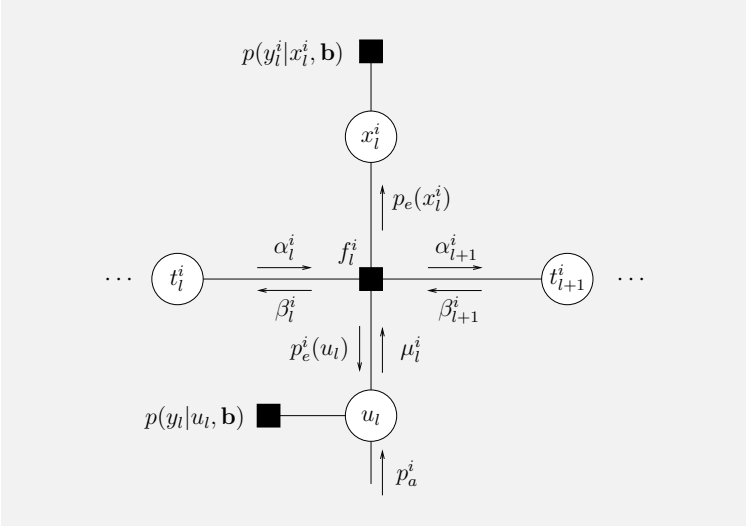

Figure 5.6 Factor-graph of one section of a constituent code. We define $f_l^i = \mathbb{I}\{t_{l+1}^i \leftarrow (u_l, t_l^i)\}\;\mathbb{I}\{x_l^i = \mathrm{out}_i(u_l, t_l^i)\}$.

Using the notations defined in Fig. 5.6, we have the forward and backward recursions of the i^{th} constituent decoder are given by

$$\alpha_l^i(t_l^i) = \sum_{\sim\{t_l^i\}} \alpha_l(t_{l-1}^i)\ \mathbb{I}\{t_l^i \leftarrow (u_{l-1}, t_{l-1}^i)\}\ \mathbb{I}\{x_{l-1}^i = \text{out}(u_{l-1}, t_{l-1}^i)\} \times p(y_{l-1}^i|x_{l-1}^i, \mathbf{b})\ p(y_{l-1}|u_{l-1}, \mathbf{b})\ p_a(u_{l-1}), \tag{5.34}$$

$$\beta_l(t_l^i) = \sum_{\sim\{t_l^i\}} \beta_{l+1}^i(t_{l+1}^i)\ \mathbb{I}\{t_{l+1}^i \leftarrow (u_l, t_l^i)\}\ \mathbb{I}\{x_l^i = \text{out}(u_l, t_l^i)\} \times\ p(y_l^i|x_l^i, \mathbf{b})\ p(y_l|u_l, \mathbf{b})\ p_a(u_l). \tag{5.35}$$

These recursions enable to compute the messages $p_e^i(u_l)$ which will be transmitted to the other constituent code as

$$p_e^i(u_l) = \sum_{\sim\{u_l\}} \alpha_l^i(t_l^i)\ \beta_{l+1}^i(t_{l+1}^i)\ \mathbb{I}\{t_{l+1}^i \leftarrow (u_l, t_l^i)\} \times \mathbb{I}\{x_l^i = \text{out}(u_l, t_l^i)\}\ p(y_l^i|x_l^i, \mathbf{b}). \tag{5.36}$$

Message $p_e^i(u_l)$ is usually referred to as *extrinsic information* since it provides information about u_l which derives from the code structure but not depends on current observation y_l. After convergence of the SP algorithm, marginals $p(u_l|\mathbf{y}, \mathbf{b})$ can be approximated as

$$p(u_l|\mathbf{y}, \mathbf{b}) \propto p(y_l|u_l, \mathbf{b})\ p_e^1(u_l)\ p_e^2(u_l), \tag{5.37}$$

$$\propto p(y_l|u_l, \mathbf{b})\ p_e(u_l), \tag{5.38}$$

where $p_e(u_l) \triangleq p_e^1(u_l)\ p_e^2(u_l)$. In the same way, a posteriori probabilities of coded bits x_l^i may be approximated as

$$p(x_l^i|\mathbf{y}, \mathbf{b}) \propto p(y_l^i|x_l^i, \mathbf{b})\ p_e(x_l^i), \tag{5.39}$$

where

$$p_e(x_l^i) = \sum_{\sim\{x_l^i\}} \alpha_l^i(t_l^i)\ \beta_{l+1}^i(t_{l+1}^i)\ \mathbb{I}\{t_{l+1}^i \leftarrow (u_l, t_l^i)\} \times \mathbb{I}\{x_l^i = \text{out}(u_l, t_l^i)\}\ p(y_l|u_l, \mathbf{b})\ p_a(u_l). \tag{5.40}$$

■

Example 5.6 (The Turbo Demodulator [3]) :

In this example we consider the receiver deriving from the SP algorithm when the bit-to-symbol mapping is not BPSK. In that case, there is no longer a one-to-one correspondence between coded bits and transmitted symbols. The factorization of $p(\mathbf{a}|\mathbf{x})$ is therefore not trivial and has to be considered in the factor graph representation of the receiver. Denoting by $x_k^1, \dots x_k^q$ the sequence of bits mapped into symbol a_k, we have

$$p(\mathbf{a}|\mathbf{x}) = \mathbb{I}\{a_k = \varphi(x_k^1, \dots x_k^q)\}, \tag{5.41}$$

i.e. transmitted symbol a_k has to correspond to bit-to-symbol mapping output $\varphi(x_k^1, \dots x_k^q)$. Probability $p(\mathbf{y}|\mathbf{a}, \mathbf{b})$ may still be factorized as

$$p(\mathbf{y}|\mathbf{a}, \mathbf{b}) = \prod_k p(y_k|a_k, \mathbf{b}), \tag{5.42}$$

whereas the factorization of $p(\mathbf{x}|\mathbf{u})$ depends on the considered code (see Examples 5.4 and 5.5). The factor graph of $p(\mathbf{y}|\mathbf{a}, \mathbf{b})\, p(\mathbf{a}|\mathbf{x})\, p(\mathbf{x}|\mathbf{u})$ may then be represented as in Fig. 5.7. The code part has been replaced by a box for the sake of generality.

Due to the presence of the code, the factor graph contains cycles. The SP algorithm is therefore iterative and will no longer lead to the computation of exact marginals. The messages passed along the edges of the factor graph are usually updated as follows (see Fig. 5.8 for the notations): messages $\mu_{f_k \to x_k^l}$ are first updated by taking into account extrinsic probabilities $p_e(x_k^l)$ delivered by the decoder, i.e.

$$\mu_{f_k \to x_k^l}(x_k^l) = \sum_{\sim\{x_k^l\}} p(y_k|a_k, \mathbf{b})\ \mathbb{I}\{a_k = \varphi(x_k^1, \dots, x_k^q)\} \prod_{n \neq l} p_e(x_k^n). \tag{5.43}$$

Then, extrinsic probabilities $p_e(x_k^l)$ are in turn updated by a decoding operation. Since the factor graph of the code remains the same, the decoding update rules derived in Examples 5.4 or 5.5 are still valid. Note however that the decoder does no longer have access to the observation and messages $\mu_{f_k \to x_k^l}(x_k^l)$ entering the coded-bit nodes have to be used instead.

Although we are ultimately interested in the computation of information-bit posterior probabilities $p(u_l|\mathbf{y}, \mathbf{b})$, symbol a posteriori probabilities $p(a_k|\mathbf{y}, \mathbf{b})$ will be shown in Chapter 7 to be of interest in the design of CA synchronizers. Using notations from Fig. 5.8, symbol a posteriori probabilities may be approximated as

$$p(a_k|\mathbf{y}, \mathbf{b}) \propto p(y_k|a_k, \mathbf{b})\, p_e(a_k). \tag{5.44}$$

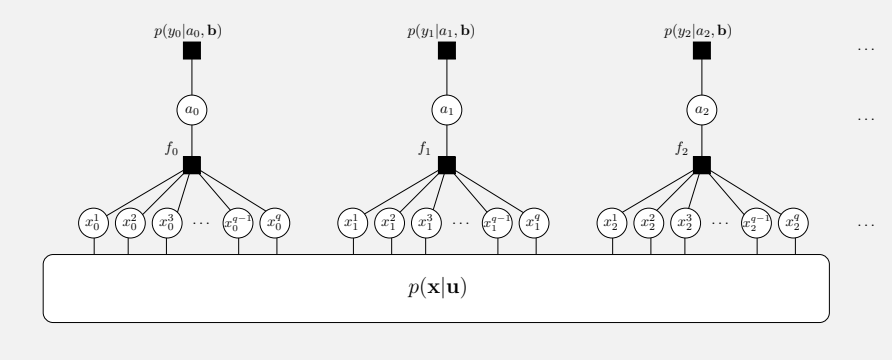

Figure 5.7 Factor-graph of the turbo demodulator. We define $f_k = \mathbb{I}\{a_k = \varphi(x_k^1, \dots x_k^q)\}$.

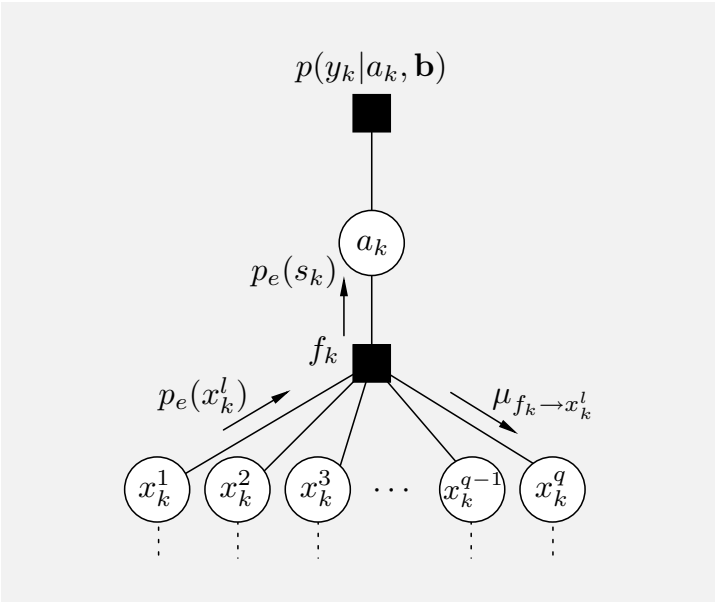

Figure 5.8 Factor-graph of one section of the demapping stage. We define $f_k = \mathbb{I}\{a_k = \varphi(x_k^1, \dots x_k^q)\}$.

$p_e(a_k)$ represents the extrinsic probability of symbol a_k and may be computed as

$$\begin{aligned} p_e(a_k) &= \sum_{\sim\{a_k\}} \left[\prod_l^q p_e(x_k^l)\ \mathbb{I}\{a_k = \varphi(x_k^1, \ldots, x_k^q)\} \right] \\ &= \prod_l^q p_e(x_k^l), \end{aligned} \tag{5.45}$$

where the last equality is valid if $a_k = \varphi(x_k^1, \ldots, x_k^q)$. The extrinsic probability of symbol a_k is therefore built by the product of the coded-bit extrinsic probabilities delivered by the decoder. ■

5.5 Main Points

In this chapter, we introduce the concepts of factor-graph representation, sum-product (SP) algorithm and extrinsic probability:

- A factor graph is a bipartite graph which graphically represents the factorization of a function (see section 5.2).
- The SP algorithm is a message-passing algorithm which applies on factor graphs (see section 5.3). If the factor graph is cycle-free, the SP algorithm enables to efficiently compute all the marginals of the function that the graph represents in a finite number of steps. If the graph contains cycles, the SP algorithm becomes iterative and is no longer able to deliver the actual marginals.
- Numerous well-known algorithms in digital communications may be regarded as particular instances of the SP algorithm. In this chapter, we replace the Viterbi algorithm [45], the BCJR algorithm [46] as well as two instances of the turbo principle [2], [3] in the context of the SP algorithm (Examples 5.4, 5.5 and 5.6 in section 5.4).
- We introduce the notion of *extrinsic probability* (Examples 5.4, 5.5 and 5.6 in section 5.4). Expressions of the extrinsic probability are given in the particular cases of a convolutionally-coded and turbo-coded transmissions with either BPSK or higher-level modulations.

5.6 Appendix

In this appendix, we show that the marginals of a function $g(v_1, v_2, ..., v_n)$ that can be represented by a cycle-free factor graph may be computed using (5.6), (5.7) and (5.8). For the sake of notational simplicity, the reasoning will be made, without loss of generality, in the semiring $(\mathbb{R}, +, \cdot)$. The proof which follows is similar to the one made in [6].

Let us assume that we want to compute marginal $g_i(v_i)$ i.e.

$$g_i(v_i) = \sum_{\sim\{v_i\}} g(v_1, v_2, ..., v_n). \tag{5.46}$$

Function $g(v_1, v_2, ..., v_n)$ can be rewritten without loss of generality as

$$g(v_1, v_2, ..., v_n) = \prod_{j=1}^{|n(v_i)|} f_j(v_i, v_{Q_j}) \tag{5.47}$$

where $|n(v_i)|$ represents the number of neighbors of node v_i and v_{Q_j} are sets of elements from $\{v_1, v_2, ..., v_n\} \backslash \{v_i\}$. Since the factor-graph representation of g is cycle-free, we have $v_{Q_j} \cap v_{Q_l} = \emptyset$ and $v_{Q_1} \cup v_{Q_2} \cup \cdots \cup v_{Q_{|n(v_i)|}} = \{v_1, v_2, ..., v_n\}$, i.e. $v_{Q_1}, v_{Q_2}, ..., v_{Q_{|n(v_i)|}}$ is a partition of $\{v_1, v_2, ..., v_n\}$. This decomposition is represented by the generic factor graph of Fig. 5.9, in which $f_1(v_i, v_{Q_1})$ is shown in expanded form.

By the distributive law and using the fact that $v_{Q_1}, v_{Q_2}, ..., v_{Q_{|n(v_i)|}}$ are pairwise disjoint, we have

$$\begin{aligned} g_i(v_i) &= \sum_{\sim\{v_i\}} \prod_{j=1}^{|n(v_i)|} f_j(v_i, v_{Q_j}), \\ &= \prod_{j=1}^{|n(v_i)|} \sum_{\sim\{v_i\}} f_j(v_i, v_{Q_j}), \end{aligned} \tag{5.48}$$

i.e. the marginal of g with respect to v_i is equal to the product of marginals of $f_j(v_i, v_{Q_j})$ with respect to v_i. Let us the consider the computation of $\sum_{\sim\{v_i\}} f_1(v_i, v_{Q_1})$. Again, without loss of generality, $f_1(v_i, v_{Q_1})$ may rewritten as

$$f_1(v_i, v_{Q_1}) = h_0(v_i, v_1, ..., v_L) \prod_{l=1}^{L} h_l(v_i, v_{Q'_l}), \tag{5.49}$$

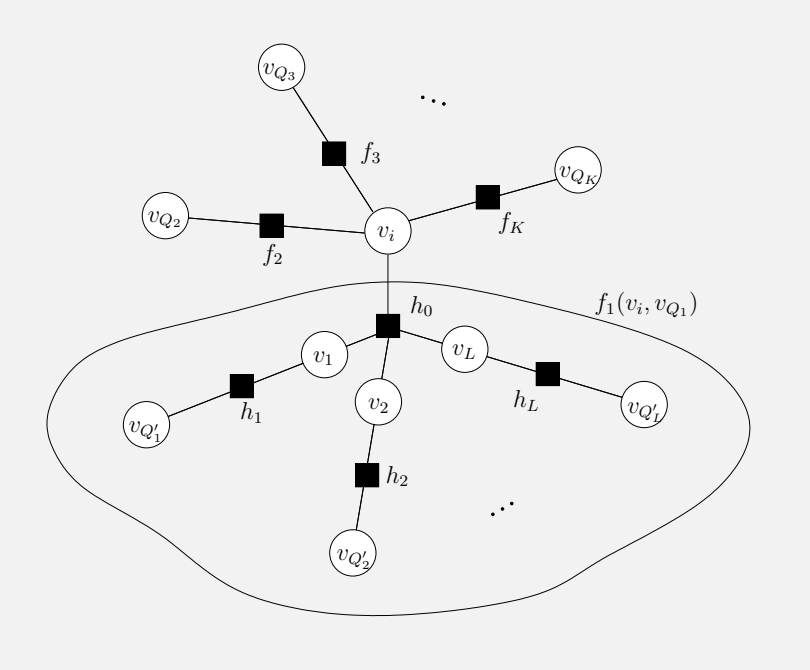

Figure 5.9 Generic cycle-free factor graph.

where, for convenience, we have numbered the arguments of g such that $v_{Q_1} = \{v_1, v_2, ..., v_L\}$. Note that $\{v_1, v_2, ..., v_n\}$, $v_{Q'_1}$,..., $v_{Q'_L}$ is a partition of v_{Q_1}. Using then the fact that these sets are pairwise disjoint and applying the distributive law, we have

$$\begin{aligned}\sum_{\sim\{v_i\}} h_1(v_i, v_{Q_1}) &= \sum_{\sim\{v_i\}} h_0(v_i, v_1, ..., v_L) \prod_{l=1}^{L} h_l(v_i, v_{Q'_l}), \\ &= \sum_{v_1,...,v_L} h_0(v_i, v_1, ..., v_L) \left(\sum_{v_{Q'_1}} h_1(v_i, v_{Q'_1})\right) \cdots \left(\sum_{v_{Q'_L}} h_L(v_i, v_{Q'_L})\right), \\ &= \sum_{\sim\{v_i\}} \left(h_0(v_i, v_1, ..., v_L) \prod_{l=1}^{L} \sum_{\sim\{v_l\}} h_l(v_l, v_{Q'_l}).\right) \end{aligned} \tag{5.50}$$

We may then observe that (5.48) and (5.50) are equivalent to (5.6), (5.7) and (5.8). Note that the above development is valid for any v_i. Therefore, updating factor-graph messages according to (5.6) and (5.7) enables to compute *all* the marginals $g_i(v_i)$ via (5.8).

Iterative Optimization Algorithms

6

In this chapter, we consider general procedures enabling to solve optimization problems such as

$$\mathbf{b}_{max} = \arg\max_{\tilde{\mathbf{b}} \in \mathcal{B}} \; g(\mathbf{b} = \tilde{\mathbf{b}}).$$

This kind of optimization problems naturally occurs in the context of estimation via the MAP and ML estimators (see Chapter 3). Unfortunately, the existence of a simple analytical solution to these problems is quite seldom and, in practice, one often has to resort to iterative algorithms to solve them.

Two essential questions can be addressed when considering iterative algorithms. The first question is whether a given algorithm yields, at least at the limit, a solution to the original problem. The second question is related to how fast an iterative algorithm converges to a solution. These questions are treated in this chapter. In particular, we first state conditions which ensure convergence to a solution. We then address the question of the speed of convergence of an iterative algorithm. Finally, we analyze two well-known kind of optimization methods, namely the gradient methods [47] and the expectation-maximization (EM) algorithm [48], which will prove to be useful in the sequel.

6.1 Convergence of Iterative Descent Methods

The different algorithms which will be considered in the sequel have the common feature to be *iterative descent algorithms*. By *iterative*, we mean that the algorithm generates a series of points, each point being calculated on the ba-

sis of the point preceding it. By *descent*, we mean that, as each new point is generated by the algorithm, the corresponding value of a function (evaluated at the most recent point) decreases in value. In the first part of this section, we give sufficient conditions for global convergence of an iterative descent algorithm. In a second part, we address the question of the speed of convergence of iterative algorithms.

6.1.1 Global Convergence Theorem

An iterative algorithm is typically initiated by specifying a starting point. If for an arbitrary starting point, the algorithm is ensured to converge to a solution, the algorithm is said *globally convergent*. Unfortunately, not all iterative algorithms have this desirable feature. The *Global Convergence Theorem* derived by Zangwill [49] gives sufficient conditions for global convergence of iterative descent algorithms:

Theorem 2 (Global Convergence Theorem) *: Let $M(\mathbf{b})$ be an algorithm on $\mathcal{B}$ and suppose that, given an initial estimate $\mathbf{b}^{(0)}$, the sequence $\{\mathbf{b}^{(n)}\}_{n=0}^{\infty}$ is generated satisfying*

$$\mathbf{b}^{(n+1)} = M(\mathbf{b}^{(n)}).$$

Let a solution set $\Gamma \subset \mathcal{B}$ be given and suppose that

i) all points $\mathbf{b}^{(n)}$ are contained in a compact set $\mathcal{B}' \subset \mathcal{B}$.

ii) there is a continuous function $f(\mathbf{b})$ on $\mathcal{B}$ such that

$$f(\mathbf{b}^{(n+1)}) < f(\mathbf{b}^{(n)}) \quad \text{if } \mathbf{b}^{(n)} \notin \Gamma,$$
$$f(\mathbf{b}^{(n+1)}) \leq f(\mathbf{b}^{(n)}) \quad \text{if } \mathbf{b}^{(n)} \in \Gamma.$$

iii) $M(\mathbf{b})$ is a continuous function on $\mathcal{B}$.

Then the limit of any sequence $\{\mathbf{b}^{(n)}\}_{n=0}^{\infty}$ belongs to the solution set Γ.

The Global Convergence Theorem specifies a precise procedure for proving convergence. First, the map M and the solution set Γ have to be determined. Then, a descent function f must be specified. Note that f has not necessarily to be equal to objective function g but can be any function satisfying condition $ii)$. Using conditions $i)$, $ii)$ and $iii)$ of the Global Convergence Theorem, we are ensured of the convergence of any sequence generated by an algorithm M to one point of a solution set Γ.

6.1.2 Speed of Convergence

The study of the speed of convergence enables to predict the relative effectiveness of different iterative algorithms. More precisely, it intends to answer the question of how fast will an iterative algorithm converge to a solution.

The speed of convergence of an iterative algorithm is usually defined in terms of *order of convergence* and *rate of convergence*. Considering the sequence of real numbers $\{b^{(n)}\}_{n=0}^{\infty}$ defined by an algorithm M and converging to a limit point $b^\star$, the order of convergence is defined as the supremum of the nonnegative numbers p satisfying

$$0 \leq \lim_{n\to\infty} \frac{\|b^{(n+1)} - b^\star\|}{\|b^{(n)} - b^\star\|^p} < \infty, \tag{6.1}$$

whereas the rate of convergence C is defined as the limit

$$C = \lim_{n\to\infty} \frac{\|b^{(n+1)} - b^\star\|}{\|b^{(n)} - b^\star\|^p}. \tag{6.2}$$

Larger values of the order of convergence p imply, in a sense, faster convergence. Indeed, we have asymptotically that

$$\|b^{(n+1)} - b^\star\| = C\,\|b^{(n)} - b^\star\|^p, \tag{6.3}$$

i.e. order p implies that the distance from limit point $b^\star$ is reduced by the p^{th} power in a single step. When iterative methods have the same order of convergence, their relative effectiveness is usually compared via their rate of convergence. From (6.3), it is clear that smaller rates of convergence imply faster convergence.

It should be noted that the order and the rate of convergence are asymptotical notions, i.e. they are in principle only valid for $n \to \infty$. In practice however, these notions may be shown to be useful to characterize non-asymptotical effectiveness of iterative algorithms and will therefore be used as such in the sequel. Note also that notions defined here for a scalar sequence $\{b^{(n)}\}_{n=0}^{\infty}$ may be extended to the vector case, e.g. see [47].

6.2 Gradient Methods

Gradient methods are based on the simple fact that sequences defined as

$$\mathbf{b}^{(n+1)} = \mathbf{b}^{(n)} + \alpha^{(n)} \nabla g(\mathbf{b}^{(n)}) \qquad \text{with } \alpha^{(n)} > 0, \tag{6.4}$$

increase objective function $g(\mathbf{b})$ as long as factor $\alpha^{(n)}$ is properly chosen. Indeed, from the first-order Taylor series expansion around $\mathbf{b}^{(n)}$ we have

$$g(\mathbf{b}^{(n+1)}) \simeq g(\mathbf{b}^{(n)}) + \nabla g(\mathbf{b}^{(n)})(\mathbf{b}^{(n+1)} - \mathbf{b}^{(n)}). \tag{6.5}$$

Plugging (6.4) into (6.5), we get

$$g(\mathbf{b}^{(n+1)}) \simeq g(\mathbf{b}^{(n)}) + \alpha \|\nabla g(\mathbf{b}^{(n)})\|^2. \tag{6.6}$$

Approximation (6.6) is valid as long as $\alpha^{(n)}$ is small enough and we can therefore conclude that there exists some $\alpha^{(n)} > 0$ such that

$$g(\mathbf{b}^{(n+1)}) \geq g(\mathbf{b}^{(n)}). \tag{6.7}$$

Inequality (6.7) may even be strict provided that $\nabla g(\mathbf{b}^{(n)}) \neq 0$: $-g(x)$ is therefore a descent function for the algorithm (6.4). The associated solution set may be defined as

$$\Gamma_{Gradient} = \left\{ \mathbf{b} : \nabla g(\mathbf{b}) = 0 \right\}, \tag{6.8}$$

i.e. $\Gamma_{Gradient}$ contains all the stationary points of $g(\mathbf{b})$ (maxima, saddle points and minima). Note that, as long as (6.7) is verified, local minima are rather fixed points than convergence points of the gradient algorithms: the method will not converge to a minimum but may nevertheless get stuck if initialized at a minimum.

Algorithm (6.4) may be generalized as

$$\mathbf{b}^{(n+1)} = \mathbf{b}^{(n)} + \alpha^{(n)} \mathbf{d}^{(n)} \qquad \text{with } \alpha^{(n)} > 0, \tag{6.9}$$

where $\mathbf{d}^{(n)}$ has to be such that

$$\nabla^T g(\mathbf{b}^{(n)})\, \mathbf{d}^{(n)} > 0, \tag{6.10}$$

i.e. such that the angle between update direction $\mathbf{d}^{(n)}$ and gradient $\nabla g(\mathbf{b}^{(n)})$ is smaller than 90 degrees. Condition (6.10) actually ensures that the update direction locally increases the objective function $g(\mathbf{b})$. Direction $\mathbf{d}^{(n)}$ is usually defined as

$$\mathbf{d}^{(n)} = \mathbf{D}^{(n)}\, \nabla g(\mathbf{b}^{(n)}). \tag{6.11}$$

The choice of $\mathbf{D}^{(n)}$ leads to different gradient methods. In the sequel, we consider in particular the Steepest-Descent (SD) and the Newton-Raphson (NR) algorithms.

6.2.1 The Steepest-Descent Algorithm

The Steepest-Descent (SD) algorithm defines matrix $\mathbf{D}^{(n)}$ as

$$\mathbf{D}^{(n)} \triangleq \mathbf{I}_{N\times N} \qquad \text{for } n = 0, 1, \ldots \tag{6.12}$$

At each iteration, $\mathbf{b}^{(n)}$ is therefore updated in the direction of the steepest slope, i.e.

$$\mathbf{b}^{(n+1)} = \mathbf{b}^{(n)} + \alpha^{(n)} \nabla g(\mathbf{b}^{(n)}). \tag{6.13}$$

The speed of convergence of the SD algorithm is *linear* and depends on the choice of factor $\alpha^{(n)}$. The rate of convergence of the SD algorithm (in the case of a scalar parameter b) is given by

$$C = 1 - \alpha^{(n)} \nabla^2 g(b) \qquad \text{where } b \in \Gamma_{Gradient}. \tag{6.14}$$

We see that the choice of factor $\alpha^{(n)}$ is crucial for the convergence of the steepest descent algorithm. On the one hand, too small $\alpha^{(n)}$ will lead to a convergence rate close to 1 and the speed of convergence will therefore be very slow. On the other hand, too large $\alpha^{(n)}$ no longer ensures the algorithm to converge. The choice of $\alpha^{(n)}$ is therefore a trade-off between the speed of convergence and the robustness of the method. Several rules exist for the choice of $\alpha^{(n)}$, see [47] for more details.

6.2.2 The Newton-Raphson Algorithm

The Newton-Raphson (NR) method defines matrix $\mathbf{D}^{(n)}$ as

$$\mathbf{D}^{(n)} = -\left(\nabla^2 g(\mathbf{b}^{(n)})\right)^{-1}, \tag{6.15}$$

so that, at each iteration, $\mathbf{b}^{(n)}$ is corrected proportionally to the first derivative and the inverse of the second derivative of $g(\mathbf{b})$, i.e.

$$\mathbf{b}^{(n+1)} = \mathbf{b}^{(n)} - \alpha^{(n)} \left(\nabla^2 g(\mathbf{b}^{(n)})\right)^{-1} \nabla g(\mathbf{b}^{(n)}). \tag{6.16}$$

When $\alpha^{(n)} = 1$, the speed of convergence of the NR algorithm is *quadratic*. This means that the NR method may find the maximum of a negative definite quadratic function in a single iteration. This ultimately fast convergence speed is often regarded as the major strength of the NR algorithm. This increase of the convergence speed is however at the expense of a recalculation

of the Hessian matrix $\nabla^2 g(\mathbf{b})$ at each iteration.

Note that the NR algorithm may converge to minima of $g(\mathbf{b})$ as well. Indeed, in the neighborhood of a minimum, Hessian matrix $\nabla^2 g(\mathbf{b})$ is positive definite (therefore so is $\left(\nabla^2 g(\mathbf{b})\right)^{-1}$) and condition (6.10) no longer holds:

$$\begin{aligned}\nabla^T g(\mathbf{b})\, \mathbf{d}^{(n)} &= -\nabla^T g(\mathbf{b}) \left(\nabla^2 g(\mathbf{b})\right)^{-1} \nabla g(\mathbf{b}), \\ &< 0.\end{aligned}$$

In this case, the NR algorithm will rather converge to a minimum. In fact, the ultimate convergence point of the NR algorithm depends on the initial point: if $\mathbf{b}^{(0)}$ is in the "attraction area" of a maximum it will converge to this maximum. Otherwise, it may converge to either a minimum or a saddle point.

6.3 EM-based Algorithms

The expectation-maximization (EM) algorithm, first defined by Dempster, Laird and Rubin in [48], is an iterative method which enables to solve ML optimization problem i.e.

$$\hat{\mathbf{b}}_{ML} = \arg\max_{\tilde{\mathbf{b}}} p(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}}). \tag{6.17}$$

In the next subsection, we give the general expressions and the main features of the EM algorithm. In a second subsection, we introduce a more general class of algorithms known as generalized EM (GEM) algorithms.

6.3.1 The Expectation-Maximization Algorithm

The mapping M performed by the EM algorithm may be divided in two main steps: the expectation step (E-step) and the maximization step (M-step). Formally, they may be expressed as

$$\text{E-step:} \qquad \mathcal{Q}(\mathbf{b}, \mathbf{b}^{(n)}) = \int_{\mathcal{Z}(\mathbf{r})} p(\mathbf{z}|\mathbf{r}, \mathbf{b}^{(n)}) \, \log \, p(\mathbf{z}|\mathbf{b}) \, d\mathbf{z}, \tag{6.18}$$

$$\text{M-step:} \qquad \mathbf{b}^{(n+1)} = \arg\max_{\tilde{\mathbf{b}}} \mathcal{Q}(\mathbf{b} = \tilde{\mathbf{b}}, \mathbf{b}^{(n)}), \tag{6.19}$$

where $\mathcal{Z}(\mathbf{r}) = \{\mathbf{z} : \mathbf{r} = f(\mathbf{z})\}$ is an alternative observation set defined by a many-to-one mapping f. Actual observation set $\mathcal{R}$ and "extended" observation set $\mathcal{Z}$ are usually respectively referred to as *incomplete* and *complete* data

set. The key idea of the EM algorithm is to associate to the incomplete-data problem, a complete-data problem easier to solve. For instance the chosen complete-data problem may have a close-form solution. This simpler ML estimate computation of the complete-data problem may be exploited in the M-step of the iterative computing algorithm.

Wu showed in [50] that the solution set of the EM algorithm is the set of stationary points of objective function $p(\mathbf{r}|\mathbf{b})$, i.e.

$$\Gamma_{EM} = \left\{ \mathbf{b} : \nabla p(\mathbf{r}|\mathbf{b}) = 0 \right\}. \tag{6.20}$$

The point of convergence of the EM algorithm could therefore be any mimimum, maximum or saddle point of objective function $p(\mathbf{r}|\mathbf{b})$. Note however that minima are rather "fixed" points than "convergence" points: the EM algorithm can be stuck at a minimum as long as it is initiated at this value; otherwise it always converges to a maximum or a saddle point of $p(\mathbf{r}|\mathbf{b})$.

The EM algorithm has a *linear* order of convergence, i.e. $p = 1$. Its rate of convergence may be expressed in the case of a scalar[1] parameter b as

$$C(\mathbf{r}) = I_c^{-1}(\mathbf{r})\, I_m(\mathbf{r}), \tag{6.21}$$

where $I_c(\mathbf{r})$ and $I_m(\mathbf{r})$ are respectively the information matrices relative to the complete data and the missing data i.e.

$$I_c(\mathbf{r}) \triangleq -\left(\int_{\mathbf{z}} p(\mathbf{z}|\mathbf{r},\tilde{b})\, \frac{\partial^2}{\partial \tilde{b}^2} \log p(\mathbf{z}|\tilde{b})\, d\mathbf{z} \right)_{\Big|\tilde{b}=\hat{b}_{ML}} \tag{6.22}$$

$$I_m(\mathbf{r}) \triangleq -\left(\int_{\mathbf{z}} p(\mathbf{z}|\mathbf{r},\tilde{b})\, \frac{\partial^2}{\partial \tilde{b}^2} \log p(\mathbf{z}|\mathbf{r},\tilde{b})\, d\mathbf{z} \right)_{\Big|\tilde{b}=\hat{b}_{ML}}. \tag{6.23}$$

Matrix I_m may actually be seen as a measure of the amount of *missing information* i.e. a measure of the difference between the amount of information contained in the complete data set and the incomplete data set [51]. We note therefore that if the missing data contains only a few information about b, the EM rate will be close to zero and the convergence fast. On the contrary, when the missing data brings a lot of relevant information about parameter b, the EM algorithm may have a very low speed of convergence. This behavior is

[1] Extension to the vector case may be found in [51].

one of the main drawback of the EM algorithm.

Despite of its slow numerical convergence in some cases, the EM algorithm has become a very popular computational method in statistics. One of the main reasons motivating its use is that, on the contrary of the general experience in numerical optimization, the implementation of the E-step and M-step is easy for many statistical problems, thanks to the nice form of the complete-data likelihood function $p(\mathbf{z}|\mathbf{b})$.

6.3.2 The Generalized EM Algorithm

The EM algorithm may actually be viewed as a particular case of a more general family of algorithms which are based on the fact that (see appendix)

$$\mathcal{Q}(\mathbf{b} = \mathbf{b}^{(n+1)}, \mathbf{b}^{(n)}) > \mathcal{Q}(\mathbf{b} = \mathbf{b}^{(n)}, \mathbf{b}^{(n)}) \tag{6.24}$$

$$\Downarrow$$

$$p(\mathbf{r}|\mathbf{b} = \mathbf{b}^{(n+1)}) > p(\mathbf{r}|\mathbf{b} = \mathbf{b}^{(n)}). \tag{6.25}$$

The algorithms which ensure (6.24)-(6.25) as long as $\mathbf{b}^{(n)} \notin \Gamma_{EM}$ are referred to as the family of the *generalized EM* (GEM) algorithms. From the Global Convergence Theorem (see section 6.1.1), we may note that GEM algorithms have the same convergence points as the EM algorithm. Well-known instances of GEM algorithms are the expectation conditional-maximization (ECM) algorithm proposed by Meng and Rubin in [52] and the space-alternating generalized expectation-maximization (SAGE) algorithm introduced by Fessler and Hero [53].

The ECM algorithm proposes to replace the M-step, i.e. the maximization of the $\mathcal{Q}$-function over the whole parameter space $\mathcal{B}$, by several conditional maximizations over subspaces $\mathcal{B}_1$,...,$\mathcal{B}_L$ of $\mathcal{B}$ (with $\mathcal{B}_1 \cup \ldots \cup \mathcal{B}_L = \mathcal{B}$). The ECM algorithm is actually well-suited to situations where the M-step remains difficult whereas conditional maximizations over smaller spaces have simple solutions. The counterpart of this simplification is a usually smaller speed of convergence, in terms of iterations, than the EM algorithm [51]. The ECM algorithm can however be faster in total computer time since conditional maximizations performed by the ECM algorithm may be simpler, faster and more stable than the corresponding full maximization performed by the EM algorithm.

As mentioned in the previous subsection, a drawback of the EM-based algorithms relies in their slow speed of convergence when the complete-data set is not sufficiently informative (see (6.21)). To cope with this problem, Fessler and Hero [53] proposed an extension of the ECM algorithm in which the definition of the complete data set may vary for each conditional maximization[2], namely the SAGE algorithm.

6.4 Main Points

In this section, we present general frameworks for the resolution of maximization problems:

- We focus on *iterative descent* algorithms i.e. algorithms which generate a sequence of points decreasing some objective function. The *Global Convergence Theorem* (section 6.1.1) provides sufficient conditions for the converge of descent algorithms to a point of a solution set.
- Gradient algorithms (section 6.2) are instances of iterative descent algorithms. Two particular cases of gradient algorithms are considered: the steepest-descent (SD) algorithm and the Newton-Raphson (NR) algorithm.
- The steepest-descent (SD) algorithm (section 6.2.1) generates a sequence of points which converges either to a maximum or a saddle point of the objective function. This algorithm exhibits a *linear* speed of convergence and its rate of convergence depends on a stepsize factor α.
- The Newton-Raphson (NR) algorithm (section 6.2.2) converges to either a maximum, a saddle point or a minimum of the objective function. Its convergence order is *quadratic*. It requires at each iteration the evaluation of the Hessian matrix of the objective function.
- Generalized Expectation-Maximization (GEM) algorithms (section 6.3.2) is another family of descent algorithms which apply in the context of maximum-likelihood (ML) estimation. Unlike gradient algorithms, GEM algorithms do not require the evaluation of the gradient of the objective function. GEM algorithms converge to either a maximum or a saddle point of the objective function with a *linear* speed of convergence. The

[2]See also the Alternating ECM algorithm proposed by Meng and van Dyk in [54]

Expectation-Maximization (EM) algorithm (section 6.3.1) is a particular instance of GEM algorithm.

6.5 Appendix

In this appendix we prove that (6.24) implies (6.25) i.e. an increase of the $\mathcal{Q}$-function leads to an increase of $p(\mathbf{r}|\mathbf{b})$. Since the logarithm is a monotonically increasing function, we can equivalently show that increasing the $\mathcal{Q}$-function increases $\log\, p(\mathbf{r}|\mathbf{b})$.

Let us first rewrite $\log\, p(\mathbf{r}|\mathbf{b})$ as

$$\log\, p(\mathbf{r}|\mathbf{b}) = \log\, p(\mathbf{r}|\mathbf{z},\mathbf{b}) + \log\, p(\mathbf{z}|\mathbf{b}) - \log\, p(\mathbf{z}|\mathbf{r},\mathbf{b}). \tag{6.26}$$

By definition of the complete data, there is a many-to-one mapping f such that $\mathbf{r} = f(\mathbf{z})$ i.e. $\mathbf{r}$ is univoquely defined by the knowledge of $\mathbf{z}$. As a consequence $p(\mathbf{r}|\mathbf{z},\mathbf{b}) = 1$ and we have

$$\log\, p(\mathbf{r}|\mathbf{b}) = \log\, p(\mathbf{z}|\mathbf{b}) - \log\, p(\mathbf{z}|\mathbf{r},\mathbf{b}). \tag{6.27}$$

Taking the expectation with respect to $p(\mathbf{z}|\mathbf{r},\mathbf{b}')$ of the two sides of (6.27), we get

$$\begin{aligned}\log\, p(\mathbf{r}|\mathbf{b}) = &\int p(\mathbf{z}|\mathbf{r},\mathbf{b}')\, \log\, p(\mathbf{z}|\mathbf{b})\, d\mathbf{z} \\ &- \int p(\mathbf{z}|\mathbf{r},\mathbf{b}')\, \log\, p(\mathbf{z}|\mathbf{r},\mathbf{b})\, d\mathbf{z}.\end{aligned} \tag{6.28}$$

where the first term in the right-hand side of (6.28) is actually equal to the $\mathcal{Q}$-function defined in (6.18).

Let us now consider $\log\, p(\mathbf{r}|\mathbf{b}^{(n+1)}) - \log\, p(\mathbf{r}|\mathbf{b}^{(n)})$ with $\mathbf{b}' = \mathbf{b}^{(n)}$:

$$\begin{aligned}\log\, p(\mathbf{r}|\mathbf{b}^{(n+1)}) - \log\, p(\mathbf{r}|\mathbf{b}^{(n)}) = &\mathcal{Q}(\mathbf{b}^{(n+1)},\mathbf{b}^{(n)}) - \mathcal{Q}(\mathbf{b}^{(n)},\mathbf{b}^{(n)}) \\ &+ \int p(\mathbf{z}|\mathbf{r},\mathbf{b}^{(n)})\, \log\, \frac{p(\mathbf{z}|\mathbf{r},\mathbf{b}^{(n)})}{p(\mathbf{z}|\mathbf{r},\mathbf{b}^{(n+1)})}\, d\mathbf{z}.\end{aligned} \tag{6.29}$$

The integral in (6.29) is a Kullback-Leibler distance [29] and therefore

$$\int p(\mathbf{z}|\mathbf{r},\mathbf{b}^{(n)})\, \log\, \frac{p(\mathbf{z}|\mathbf{r},\mathbf{b}^{(n)})}{p(\mathbf{z}|\mathbf{r},\mathbf{b}^{(n+1)})}\, d\mathbf{z} \geq 0. \tag{6.30}$$

Using (6.29) and (6.30), we have therefore that

$$\begin{aligned} &\mathcal{Q}(\mathbf{b}^{(n+1)}, \mathbf{b}^{(n)}) > \mathcal{Q}(\mathbf{b}^{(n)}, \mathbf{b}^{(n)}) \\ &\qquad\qquad \Downarrow \\ &\log\, p(\mathbf{r}|\mathbf{b}^{(n+1)}) > \log\, p(\mathbf{r}|\mathbf{b}^{(n)}). \end{aligned}$$

Code-Aided Synchronization of Continuous Parameters

7

In this chapter, we address the problem of code-aided synchronization of continuous parameters[1]. The parameters of interest are therefore (see chapter 2) the channel gain A, the fractional delay ϵ_τ, the carrier frequency offset ν and the fractional carrier phase offset ϵ_θ. We assume that $k_\theta = 0$ and $k_\tau = 0$ so that $\theta = \epsilon_\theta$ and $\tau = \epsilon_\tau$.

This chapter is organized as follows. In the first section, we motivate the use of the ML estimator for continuous parameter estimation and emphasize some issues related to its practical implementation. In the second section, we consider the classical approximations leading to conventional DA and NDA synchronizers. In the third section we introduce and motivate the concept of *soft-decision-directed* synchronization. The remainder of the chapter is then dedicated to the presentation of general mathematical frameworks enabling a systematic derivation of CA ML synchronizers operating in a soft-decision-directed mode:

- In section 7.4, we apply the EM algorithm (see chapter 6) to the problem of CA ML synchronization. After having derived the main equations of the EM synchronizer, we consider the problems related to its implementation.
- In section 7.5 we show how to solve the CA ML maximization problem

[1] We also refer the interested reader to the excellent work by Wymeersch [19].

by means of gradient optimization methods (see chapter 6). In particular, expressions allowing an easy computation of the first and second derivatives of the objective function are derived.

- In section 7.6, we show that the EM and the gradient frameworks may be combined to lead to an hybrid synchronization framework. In particular, we emphasize that the synchronizers deriving from that hybrid framework have the same points of convergence than the synchronizers derived from the EM or the gradient frameworks.
- In section 7.7, we replace the CA iterative synchronization into the framework of the factor-graph representation and the SP algorithm (see chapter 5). The frameworks based on the EM algorithm and the gradient methods derived in section 7.4 and 7.5 are then combined to the SP-based framework to give very efficient and broadly-applicable CA synchronization methods.

7.1 ML Synchronization

In this chapter, we consider the problem of designing "good" estimators of continuous synchronization parameters. As mentioned in chapter 3, the performance of continuous parameter estimators is often assessed in terms of mean square error (MSE). The optimal estimator in that sense is the MMSE estimator (3.6) which minimizes by design the MSE achieved by the estimator. Unfortunately, this estimator requires a multi-dimensional integration over the synchronization parameter space and is therefore usually quite cumbersome to implement. Instead, we will consider maximum-a-posteriori (MAP) estimation, i.e.

$$\begin{aligned}\hat{\mathbf{b}}_{MAP} &= \arg\max_{\tilde{\mathbf{b}}} \log p(\mathbf{b}=\tilde{\mathbf{b}}|\mathbf{r}),\\ &= \arg\max_{\tilde{\mathbf{b}}} \Big\{\log p(\mathbf{r}|\mathbf{b}=\tilde{\mathbf{b}}) + \log p(\mathbf{b}=\tilde{\mathbf{b}})\Big\} \qquad (7.1)\end{aligned}$$

which is slightly simpler to implement while exhibiting performance close to the MMSE estimator as long as the number of observations is large enough (see chapter 3). Since in practice we usually do not have any a priori information about the synchronization parameters, $p(\mathbf{b})$ is assumed uniform and MAP estimation reduces to ML estimation, i.e.

$$\hat{\mathbf{b}}_{ML} = \arg\max_{\tilde{\mathbf{b}}} \log p(\mathbf{r}|\mathbf{b}=\tilde{\mathbf{b}}), \qquad (7.2)$$

where

$$p(\mathbf{r}|\mathbf{b}) = \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{b},\mathbf{a})\, p(\mathbf{a}). \tag{7.3}$$

ML maximization problem (7.2) does in general not have any explicit solution. In practice, approximations of the actual ML solution are therefore usually considered. In the next section, we review the common approximations made to derive conventional synchronization methods [8, 9].

7.2 Conventional Suboptimal Approaches

As mentioned in section 7.1, ML maximization problem (7.2) does not have an analytical solution. Consequently, instead of computing the exact ML estimate, conventional synchronizers proposed in the literature [8, 9] are rather smart approximations of the true ML solution. Conventional synchronizers may be divided according to the approximation they make about the sequence a priori distribution $p(\mathbf{a})$ (see chapter 4). We distinguish between data-aided (DA) and non-data-aided (NDA) synchronizers.

7.2.1 Data-Aided Synchronizers

A first approach to find an analytical solution to (7.2) consists in transmitting a sequence of K_{pil} known symbols (pilots), say $\mathbf{a_{pil}}$, and assuming that $p(\mathbf{a})$ is distributed as

$$p(\mathbf{a}) = \delta(\mathbf{a} - \tilde{\mathbf{a}}) \qquad \text{where } \tilde{\mathbf{a}} \triangleq [\mathbf{a_{pil}}^T\, \mathbf{0}_{K-K_{pil}}^T]^T. \tag{7.4}$$

The transmitted sequence is therefore assumed to be perfectly-known and the data symbols are assumed to be equal to zero. Plugging approximation (7.4) into (7.3), we get

$$\hat{\mathbf{b}}_{DA} = \arg\max_{\tilde{\mathbf{b}}} \log p(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}}, \mathbf{a} = \tilde{\mathbf{a}}). \tag{7.5}$$

$p(\mathbf{r}|\mathbf{b},\mathbf{a})$ is a Gaussian distribution and the resolution of the maximization problem (7.5) leads to an explicit solution:

$$[\hat{\nu}_{DA},\hat{\tau}_{DA}] = \arg\max_{\tilde{\nu},\tilde{\tau}}\left\{\left\|\sum_{k=0}^{K_{pil}-1} a_k^* \, y_k(\tilde{\nu},\tilde{\tau})\right\|\right\}, \tag{7.6}$$

$$\hat{\theta}_{DA} = \arg\left\{\sum_{k=0}^{K_{pil}-1} a_k^* \, y_k(\hat{\nu}_{DA},\hat{\tau}_{DA})\right\}, \tag{7.7}$$

$$\hat{A}_{DA} = \frac{\left\|\sum_{k=0}^{K_{pil}-1} a_k^* \, y_k(\hat{\nu}_{DA},\hat{\tau}_{DA})\right\|}{\sum_{k=0}^{K_{pil}-1}\|a_k\|^2}, \tag{7.8}$$

where $y_k(\nu,\tau)$ denotes the matched-filter output computed at time $kT+\tau$ and assuming carrier frequency offset ν (see definition (3.27)). From (7.6)-(7.8), we see that the synchronization parameter estimates are only based on the first K_{pil} matched-filter outputs. This method is therefore clearly suboptimal since the part of the observations related to the data symbols is not taken into account in the computation of the estimates. As a consequence an increase of the estimation quality also implies an increase of either the number of transmitted pilots or the power allocated to the pilot sequence; both solutions leading to a waste of resources. In practice, the solutions based on the transmission of pilots should therefore be avoided as much as possible.

7.2.2 Non-Data-Aided Synchronizers

DA approximation (7.4) leads to estimates which do not exploit all the observation set and loose therefore a part of the available information. NDA methods provide a solution to this problem by making a different assumption on the transmitted sequence: although the transmission may be coded, they assume that all the possible sequences are a priori equiprobable, i.e.

$$p(\mathbf{a}) = \frac{1}{|\mathcal{A}^K|} \qquad \forall\, \mathbf{a} \in \mathcal{A}^K. \tag{7.9}$$

Considering then some further approximations, close-form expressions for the estimation of synchronization parameters may be derived [8, 9]. NDA methods usually exhibit a low complexity and are therefore well-suited for practical implementation. However, this kind of approaches does not take into account the actual a priori information about the transmitted sequence and drops therefore some potentially-relevant information. In particular, as

emphasized in chapter 4, the degradation due to approximation (7.9) may be important at low SNRs.

We illustrate the NDA approach by two important examples: the Oerder&Meyr timing synchronizer [55] and the Viterbi&Viterbi carrier phase synchronizer [56]:

Example 7.1 (The Oerder&Meyr Timing Synchronizer [55]) :

In conventional synchronization methods, carrier phase and timing offset estimation are often considered separately [8, 9]: timing synchronization is first performed followed by carrier phase offset correction. In such a case, timing estimators are required to be phase-independent. When based on the ML criterion, one is therefore interested in solving the following maximization problem:

$$\hat{\tau} = \arg\max_{\tilde{\tau}} \log p(\mathbf{r}|\tau = \tilde{\tau}), \tag{7.10}$$

where

$$p(\mathbf{r}|\tau) = \int_{\theta} \sum_{\mathbf{a}} p(\mathbf{r}|\tau, \theta, \mathbf{a})\, p(\mathbf{a})\, p(\theta)\, d\theta. \tag{7.11}$$

Note that A and ν are assumed to be known with $A = 1$ and $\nu = 0$, whereas $p(\theta)$ is assumed to be uniform over $[-\pi, \pi]$.

The Oerder&Meyr synchronizer is a phase-independent NDA timing estimator based on a low-SNR approximation of (7.11). This approximation leads to an alternative maximization problem, i.e.,

$$\hat{\tau} = \arg\max_{\tilde{\tau}} \Big\{ \sum_{k} \parallel y_k(\tau = \tilde{\tau}, \nu = 0) \parallel^2 \Big\}, \tag{7.12}$$

whose solution is given [8, 9] by

$$\hat{\tau}_{OM} = \frac{T}{2\pi} \arg\Big\{ \sum_{m} \|y(mT_s)\|^2\, e^{-j2\pi \frac{mT_s}{T}} \Big\}. \tag{7.13}$$

where $y(mT_s)$ denotes the matched-filter output computed at time mT_s. Note that the arg-function in (7.13) may only deliver values between $-\pi$ and π and the timing estimate is then stuck in the interval $[-\frac{T}{2}, \frac{T}{2}]$. The Oerder&Meyr can therefore not recover timing offset whose magnitude is larger than half a symbol period, i.e. $|\tau| > \frac{T}{2}$. This behavior is actually due to the NDA nature of the synchronizer. Other timing estimators based on the same principle as the Oerder&Meyr may be found in [57]. ■

Example 7.2 (The Viterbi&Viterbi Carrier Phase Synchronizer [56]) :

Since conventional timing synchronization is usually performed prior to estimating the phase, conventional phase synchronizers often operate in a timing-directed mode i.e. they assume the timing to be known and equal to $\hat{\tau}$. The ML problem to solve under this assumption writes[2]

$$\hat{\theta} = \arg\max_{\tilde{\theta}} \; p(\mathbf{r}|\theta = \tilde{\theta}, \tau = \hat{\tau}), \tag{7.14}$$

where

$$p(\mathbf{r}|\theta, \tau) = \sum_{\mathbf{a}} p(\mathbf{r}|\theta, \tau, \mathbf{a})\, p(\mathbf{a}). \tag{7.15}$$

The Viterbi&Viterbi synchronizer is a NDA carrier phase estimator based on some low-SNR approximations of (7.15). It may be expressed as

$$\hat{\theta}_{VV} = \frac{1}{M} \arg\left\{ \sum_{k} y_k^M(\tau = \hat{\tau}, \nu = 0) \right\}. \tag{7.16}$$

Since the arg $-$function in (7.16) only delivers values between $-\pi$ to π, the Viterbi&Viterbi synchronizer is only able to recover phase offsets belonging to the interval $[-\frac{\pi}{M}, \frac{\pi}{M}]$. The Viterbi&Viterbi synchronizer therefore suffers from an M-fold ambiguity. This feature is due to the NDA nature of the synchronizer. ■

7.3 Implementation Modes

As mentioned in the previous section, both DA and NDA approximations lead to the loss of some statistical information: DA synchronizers do not take into account a part of the observation set while NDA synchronizers neglect some available sequence a priori information. These approximations may lead to important degradations in terms of estimation quality, especially at low SNR (see chapter 4). As a consequence, state-of-the-art receivers, which operate at very low SNR, more and more require to solve maximization problem (7.2) using the actual sequence a priori information, namely

$$\begin{aligned} p(\mathbf{a} = \tilde{\mathbf{a}}) &= \frac{1}{|\mathcal{S}|} \qquad \forall\ \tilde{\mathbf{a}} \in \mathcal{S}, \\ &= 0 \qquad \forall\ \tilde{\mathbf{a}} \notin \mathcal{S}, \end{aligned} \tag{7.17}$$

[2] A and ν are assumed to be known with $A = 1$ and $\nu = 0$.

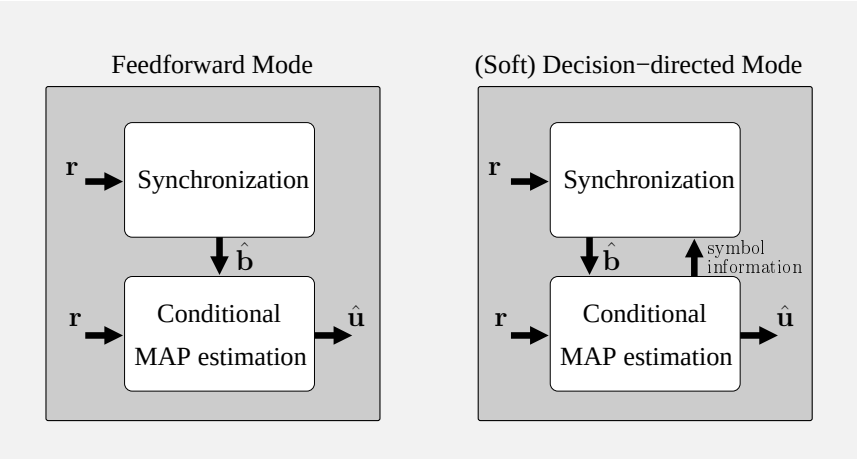

Figure 7.1 Feedforward synchronization versus iterative decision-directed synchronization.

where $\mathcal{S} \subset \mathcal{A}^K$ is the subset of sequences legitimated by the code used for the transmission.

CA ML problem does unfortunately not have any simple analytical solution. An idea proposed by several authors to deal with this problem is to provide to the synchronizer some symbol information coming from the symbol-detection part of the receiver, see e.g. [8, 9, 15, 58]. The justification of this approach rests on the observation that the gap between NDA and CA synchronizer performance is due to the fact that no symbol information is available in the NDA case. Taking benefit from some symbol information delivered by the symbol-detection part, the synchronizer is then expected to improve the estimation quality. Based on this principle, we distinguish between three main *implementation modes*:

- **Feedforward (FF) mode:** In this mode, no symbol information is delivered by the symbol detector to the synchronizer. The synchronizer simply computes an estimate and send it to the symbol detector without interacting with it. Conventional DA (see (7.6)-(7.8)) and NDA (see (7.13), (7.16)) synchronizers are examples of feedforward synchronizers.

- **Decision-directed (DD) mode:** In the DD mode, the synchronizer computes an estimate by assuming that the decision made, say $\hat{\mathbf{a}}$, by the

symbol-detection part of the receiver is correct. Considering the ML criterion, a DD implementation leads to the following maximization problem:

$$\hat{\mathbf{b}} = \arg\max_{\tilde{\mathbf{b}}} \; p(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}}, \mathbf{a} = \hat{\mathbf{a}}). \tag{7.18}$$

The resolution of this problem has a complexity comparable to the complexity of a DA problem and is therefore tractable.

- **Soft-decision-directed (SDD) mode:** In the SDD mode, the symbol-detection part of the receiver no longer feeds the synchronizer with hard symbol decisions but rather with a measure of the symbol knowledge we have after a detection/decoding operation. This measure can for example be the probability that the symbol is equal to some value of the constellation alphabet $\mathcal{A}$. The information transmitted to the synchronizer is usually referred to as *soft information* in contrast with the hard decisions delivered when operating in DD mode.

Fig. 7.1 compares the structure of a receiver based on a feedforward implementation with a receiver based on a (soft) decision-directed implementation. We see that in the latter case, there is some interactions (possibly iterative) between the synchronizer and the symbol detector.

We draw the reader attention on the fact that implementation modes should not be mixed up with synchronization modes defined in section 4.5 (namely DA, CA and NDA modes). On the one hand, synchronization modes define the kind of a priori information which is taken into account in the synchronization algorithm. The choice of the synchronization mode therefore affects the ultimate performance achievable by a synchronizer. On the other hand, implementation modes define the way in which the synchronizer is implemented but do not influence its theoretically-achievable performance.

In the remainder of this chapter we will focus more particularly on the design of SDD synchronizers. This implementation mode is especially of interest in state-of-the-art receivers which handle soft symbol information while requiring accurate synchronization parameter estimates. Several synchronization algorithms using soft symbol information have already been proposed in the literature, see e.g. [15], [58]. Although exhibiting performance improve-

ment, these algorithms are often quite ad-hoc in the sense that their design does not rely on any rigorous mathematical framework. We consider here two examples of such algorithms which will be shown to be particular cases of the frameworks derived in the remainder of the chapter:

Example 7.3 (Lottici&Luise SDD Phase Synchronizer [15]) :

Lottici and Luise propose an iterative ML-based phase estimator which makes use of some soft symbol information available from the symbol-detection part of the receiver. Although derived for a receiver embedding a turbo decoder, their approach applies in any receiver whose symbol-detection part is based on the SP algorithm (see for instance examples 5.4, 5.5 and 5.6). Formally, the phase estimator derived by these authors may be written as

$$\hat{\theta}^{(n+1)} = \arg\max_{\tilde{\theta}} \; p^{(n)}(\mathbf{r}|\,\theta = \tilde{\theta}) \tag{7.19}$$

$$= \arg\max_{\tilde{\theta}} \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a}, \theta = \tilde{\theta})\, p^{(n)}(\mathbf{a}) \tag{7.20}$$

where

$$p^{(n)}(\mathbf{a}) \propto \prod_k p(y_k|a_k, \theta = \hat{\theta}^{(n)})\, p_e(a_k), \tag{7.21}$$

and $p_e(a_k)$ denotes the extrinsic probability associated to symbol a_k (see chapter 5). The principle of this approach consists therefore in solving at each iteration an ML problem in which the a priori sequence probability has been modified. In particular, Lottici and Luise propose to use the product of factors $p(y_k|a_k, \theta = \hat{\theta}^{(n)})\, p_e(a_k)$ as a priori information[3]. These factors are often referred to as *a posteriori soft information*. Indeed, looking back at example 5.6, we note that the product of messages arriving at variable node a_k is actually equal to $p(y_k|a_k, \theta = \hat{\theta}^{(n)})\, p_e(a_k)$. Now, if the factor graph is cycle free, the product of messages entering variable node a_k is equal (up to a normalization factor) to the a posteriori probability $p(a_k|\mathbf{r}, \theta = \hat{\theta}^{(n)})$. If the graph contains cycles, $p(y_k|a_k, \theta = \hat{\theta}^{(n)})\, p_e(a_k)$ is only an approximation of $p(a_k|\mathbf{r}, \theta = \hat{\theta}^{(n)})$.

Using a low-SNR approximation, the authors come up with an analytical

[3] In fact, using notations from example 5.6, Lottici and Luise propose to use $p^{(n)}(\mathbf{a}) \propto \prod_k \prod_l \mu_{f_k \to x_k^l}(x_k^l)\, p_e(x_k^l)$. This definition is equivalent to (7.21) in the case of BPSK modulation but differs for higher modulation order. For the sake of establishing relationships between different algorithms, we will however stick to definition (7.21) in the sequel.

solution of maximization problem (7.20):

$$\hat{\theta}^{(n+1)} = \arg\Big\{ \sum_k \eta_k^*(\mathbf{r}, \hat{\theta}^{(n)})\, y_k(\tau, \nu) \Big\} \tag{7.22}$$

where

$$\eta_k(\mathbf{r}, \hat{\theta}^{(n)}) = \sum_{a \in \mathcal{A}} a \; p^{(n)}(a_k = a). \tag{7.23}$$

Note the similarity between (7.7) and (7.22): the latter is formally obtained from the former by simply replacing the actual symbols a_k by their respective (possibly approximated) a posteriori expected values $\eta_k(\mathbf{r}, \hat{\theta}^{(n)})$. These mean values are not constellation points but rather weighted averages over all constellation points according to probability $p^{(n)}(a_k)$. Averages $\eta_k(\mathbf{r}, \hat{\theta}^{(n)})$ may therefore be regarded as soft decisions about the transmitted symbols. ■

Example 7.4 (Zhang&Burr SDD Phase Synchronizer [58]) :

The approach proposed by Zhang and Burr is basically the same as the one proposed by Lottici and Luise: at each iteration, the phase estimate is updated by solving a modified ML problem like (7.20). Zhang and Burr however define $p^{(n)}(\mathbf{a})$ as

$$p^{(n)}(\mathbf{a}) = \prod_k p_e(a_k), \tag{7.24}$$

i.e. as the product of the extrinsic symbol probabilities delivered by the symbol-detection part of the receiver. A solution to maximization problem (7.20) is then found by expanding $p^{(n)}(\mathbf{r}|\theta)$ as a Fourier series. The resulting solution is however quite cumbersome to implement and will not be further detailed here. ■

We note that Lottici&Luise's and Zhang&Burr's approaches use different symbol information to implement their SDD synchronizers. In this context we may ask the question of what kind of soft information should be used and how should this soft information be used ? In the remainder of this chapter, we derive mathematical frameworks for SDD synchronization which intend to answer those questions.

7.4 Iterative Synchronization Based on the EM Algorithm

The expectation-maximization (EM) algorithm, first defined by Dempster, Laird and Rubin in [48], is a method which enables to iteratively solve ML estimation problems (see chapter 6). This method is well-suited to situations where the ML estimation of a given parameter $\mathbf{b}$ would be straightforward if some additional data was known. In the problem we are dealing with for example, the solution of maximization problem (7.2) would be straightforward if a perfect knowledge of the transmitted sequence $\mathbf{a}$ was available (see section 7.2.1). In this section, we will therefore apply the EM algorithm to the resolution of CA ML problem.

In a first part of this section, the general equations of the EM algorithm will be particularized to the ML synchronization of a bandpass linearly-modulated signal. Then, questions associated to the implementation of the EM-based synchronizer will be discussed. In particular, we will show that the phase synchronizer proposed by Lottici and Luise (see example 7.3) is an approximation of the EM synchronization framework derived in this section.

7.4.1 The EM Algorithm Applied to ML Synchronization

In this section we apply the general framework of the EM algorithm to the particular case of the synchronization of a digital data-modulated bandpass signal. Let us recall that the EM algorithm states that the sequence $\{\hat{\mathbf{b}}^{(n)}\}_{n=0}^{\infty}$ defined by

$$\mathcal{Q}(\mathbf{b}, \hat{\mathbf{b}}^{(n)}) = \int_{\mathcal{Z}(\mathbf{r})} p(\mathbf{z}|\mathbf{r}, \hat{\mathbf{b}}^{(n)}) \, \log p(\mathbf{z}|\mathbf{b}) \, d\mathbf{z}, \tag{7.25}$$

$$\hat{\mathbf{b}}^{(n+1)} = \arg\max_{\tilde{\mathbf{b}}} \Big\{ \mathcal{Q}(\mathbf{b} = \tilde{\mathbf{b}}, \hat{\mathbf{b}}^{(n)}) \Big\}, \tag{7.26}$$

converges under some general conditions [50] to $\hat{\mathbf{b}}_{ML}$. $\mathcal{Z}(\mathbf{r})$ represents the complete-data set and should ideally be defined such that the complete-data ML problem is easy to solve (see chapter 6).

In the problem we are dealing with, an explicit solution of the ML problem exists if the transmitted sequence $\mathbf{a}$ is assumed to be known (see Section 7.2.1). Let us therefore define the complete data set as $\mathcal{Z}(\mathbf{r}) = \{\mathbf{z} : \mathbf{z} = [\mathbf{r}^T, \mathbf{a}^T]^T\}$.

According to this definition, we have

$$p(\mathbf{z}|\mathbf{b}) = p(\mathbf{r}, \mathbf{a}|\mathbf{b}). \tag{7.27}$$

Using Bayes rule and taking into account the independence of $\mathbf{a}$ and $\mathbf{b}$ we may further write

$$p(\mathbf{z}|\mathbf{b}) = p(\mathbf{r}|\mathbf{a}, \mathbf{b})\, p(\mathbf{a}|\mathbf{b}) = p(\mathbf{r}|\mathbf{a}, \mathbf{b})\, p(\mathbf{a}). \tag{7.28}$$

By substituting this result into (7.25), we have that the function which has to be maximized at each iteration of the EM algorithm writes

$$\begin{aligned} \mathcal{Q}(\mathbf{b}, \hat{\mathbf{b}}^{(n)}) = &\sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n)})\ \log p(\mathbf{r}|\mathbf{a}, \mathbf{b}) \\ &+ \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n)})\ \log p(\mathbf{a}). \end{aligned} \tag{7.29}$$

The second term in (7.29) does not depend on $\mathbf{b}$ and does consequently not affect the maximization operation in (7.26). It can therefore be dropped and the $\mathcal{Q}$-function has the following structure:

$$\mathcal{Q}(\mathbf{b}, \hat{\mathbf{b}}^{(n)}) = \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n)})\ \log p(\mathbf{r}|\mathbf{a}, \mathbf{b})\ . \tag{7.30}$$

Let us now particularize (7.30) to the model of the received observations (2.8). Since the noise affecting the observations is Gaussian, we have by neglecting the terms independent of $\mathbf{b}$ that

$$\begin{aligned} \log p(\mathbf{r}|\mathbf{a}, \mathbf{b}) \propto &\frac{A}{N_0}\ \mathrm{Re}\Big\{ \sum_{k=0}^{K-1} a_k^*\, y_k(\nu, \tau)\, e^{-j\theta} \Big\} \\ &- \frac{A^2}{2N_0} \sum_{k=0}^{K-1} \|a_k\|^2, \end{aligned} \tag{7.31}$$

where $y_k(\tau, \nu)$ is the matched-filter output computed at time $kT + \tau$ and assuming frequency offset ν, see (3.27).

Let us define for each transmitted symbol a_k

$$\begin{aligned} \eta_k(\mathbf{r}, \hat{\mathbf{b}}^{(n)}) &\stackrel{\triangle}{=} \sum_{\mathbf{a} \in \mathcal{A}^K} a_k\ p(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n)}) \\ &= \sum_{a \in \mathcal{A}} a\ p(a_k = a|\mathbf{r}, \hat{\mathbf{b}}^{(n)}), \end{aligned} \tag{7.32}$$

$$\begin{aligned} \rho_k(\mathbf{r}, \hat{\mathbf{b}}^{(n)}) &\stackrel{\triangle}{=} \sum_{\mathbf{a} \in \mathcal{A}^K} \|a_k\|^2\ p(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n)}) \\ &= \sum_{a \in \mathcal{A}} \|a\|^2 p(a_k = a|\mathbf{r}, \hat{\mathbf{b}}^{(n)}). \end{aligned} \tag{7.33}$$

Using these definitions and replacing $\log p(\mathbf{r}|\mathbf{a},\mathbf{b})$ by (7.31) in (7.30), we get

$$\mathcal{Q}(\mathbf{b},\hat{\mathbf{b}}^{(n)}) \propto \frac{A}{N_0}\,\mathrm{Re}\Big\{\sum_{k=0}^{K-1}\eta_k^*(\mathbf{r},\hat{\mathbf{b}}^{(n)})\,y_k(\nu,\tau)\,e^{-j\theta}\Big\} - \frac{A^2}{2N_0}\sum_{k=0}^{K-1}\rho_k(\mathbf{r},\hat{\mathbf{b}}^{(n)}). \tag{7.34}$$

Due to the particular structure of $\mathcal{Q}(\mathbf{b},\hat{\mathbf{b}}^{(n)})$ the synchronization parameter joint maximization problem (7.26) leads to a partially-decoupled solution

$$\big[\hat{\nu}^{(n+1)},\hat{\tau}^{(n+1)}\big] = \arg\max_{\tilde{\nu},\tilde{\tau}}\Big\{\Big\|\sum_{k=0}^{K-1}\eta_k^*(\mathbf{r},\hat{\mathbf{b}}^{(n)})\,y_k(\tilde{\nu},\tilde{\tau})\Big\|\Big\}, \tag{7.35}$$

$$\hat{\theta}^{(n+1)} = \arg\Big\{\sum_{k=0}^{K-1}\eta_k^*(\mathbf{r},\hat{\mathbf{b}}^{(n)})\,y_k(\hat{\nu}^{(n+1)},\hat{\tau}^{(n+1)})\Big\}, \tag{7.36}$$

$$\hat{A}^{(n+1)} = \frac{\big\|\sum_{k=0}^{K-1}\eta_k^*(\mathbf{r},\hat{\mathbf{b}}^{(n)})\,y_k(\hat{\nu}^{(n+1)},\hat{\tau}^{(n+1)})\big\|}{\sum_{k=0}^{K-1}\rho_k(\mathbf{r},\hat{\mathbf{b}}^{(n)})}. \tag{7.37}$$

Note the similarity between the DA synchronizer expressions (7.6)-(7.8) and the EM update equations (7.35)-(7.37): the latter ones are equal to the former ones in which actual symbols have been replaced by a posteriori means $\eta_k(\mathbf{r},\hat{\mathbf{b}}^{(n)})$. This fact is actually a direct consequence of the particular choice of the complete data set.

Note also the similarity between the SDD phase synchronizer proposed by Lottici&Luise (7.22) and the phase update equation (7.36) derived from the EM algorithm framework: both expressions have exactly the same structure. The EM algorithm approach requires however the actual a posteriori probabilities $p(a_k|\mathbf{r},\hat{\mathbf{b}}^{(n)})$ to compute average values $\eta_k(\mathbf{r},\hat{\mathbf{b}}^{(n)})$ whereas Lottici&Luise rather use the product of the messages entering factor-graph node a_k, i.e. $p(y_k|a_k,\hat{\mathbf{b}}^{(n)})\,p_e(a_k)$ (see example 7.3 for more details). If the graph is cycle-free, $p(a_k|\mathbf{r},\hat{\mathbf{b}}^{(n)}) \propto p(y_k|a_k,\hat{\mathbf{b}}^{(n)})\,p_e(a_k)$ and Lottici&Luise's approach exactly implements an EM algorithm. If the graph contains cycles, then $p(y_k|a_k,\hat{\mathbf{b}}^{(n)})\,p_e(a_k)$ is only an approximation of $p(a_k|\mathbf{r},\hat{\mathbf{b}}^{(n)})$ and Lottici&Luise's phase synchronizer is only an approximation of an EM-based synchronizer.

7.4.2 The ECM Algorithm Applied to ML Synchronization

As mentioned in chapter 6, the Expectation Conditional-Maximization (ECM) algorithm is a modified version of the standard EM algorithm. Instead of maximizing the $\mathcal{Q}$-function over the overall synchronization parameter space $\mathcal{B}$, the ECM algorithm performs conditional maximizations over subspaces $\mathcal{B}_i$ (with $\bigcup_i \mathcal{B}_i = \mathcal{B}$). The ECM algorithm is part of the GEM-algorithm (see chapter 6) family and therefore benefits from the same convergence properties. In particular, it increases the log-likelihood function $\log p(\mathbf{r}|\mathbf{b})$ as long as a stationary point of $\log p(\mathbf{r}|\mathbf{b})$ has not been reached.

Since the E-step remains unchanged with respect to a conventional EM algorithm, the expression of the $\mathcal{Q}$-function derived in section 7.4.1, i.e. (7.34), is still valid in the context of ECM synchronization. The M-step is modified and replaced by several easier conditional maximizations. In particular, the joint maximum search of ν and τ in (7.35) may be replaced by two easier one-dimensional searches. The gain associated to the use of the ECM instead of the EM algorithm can appear quite poor in the simple case considered in this dissertation. However, the ECM approach can be of particular interest in more complicated synchronization problems, in which the number of synchronization parameters is large.

7.4.3 Implementation of the EM-based Synchronizer

In this section we address the implementation of the EM-based synchronizer in practical receivers. In particular, we discuss the computational complexity of the E-step and the M-step.

The maximization problem (7.26) which has to be solved at each EM iteration leads to a partially-decoupled solution, see (7.35)-(7.37). As mentioned above, those expressions are very similar to the ones of a DA synchronizer and therefore, the complexity of the M-step is similar to the complexity of a DA synchronizer. Note that the timing estimate and carrier-frequency offset estimate do not have any close-form solution but require to numerically solve a maximization problem. In more complicated synchronization problems, the M-step may be simplified by considering the ECM synchronizer defined in section 7.4.2.

Algorithm 1: Synchronization based on the EM algorithm

initialization:
$\hat{\mathbf{b}}^{(0)} = \mathbf{b}_0$;
repeat
`compute`$(p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)}))$;
`reset`(*factor-graph messages*);
`run`(*SP algorithm*);
$p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)}) \propto p(y_k|a_k, \hat{\mathbf{b}}^{(n)})\, p_e(a_k)$;
`compute`$(\hat{\mathbf{b}}^{(n+1)} = \arg\max_{\tilde{\mathbf{b}}} \mathcal{Q}(\mathbf{b} = \tilde{\mathbf{b}}, \hat{\mathbf{b}}^{(n)}))$
until *convergence*;

From (7.32) and (7.33), we note that the E-step only requires the knowledge of marginal a posteriori probabilities $p(a_k|\mathbf{r}, \mathbf{b})$. Since the M-step has a low complexity, the computation of those probabilities (and therefore the E-step) dominates the complexity of the EM synchronizer. Note that most of state-of-the-art receivers are based on CMAP estimation of the transmitted data (see chapter 4) i.e. make a decision on the transmitted symbols[4] as follows

$$\hat{a}_{k,CMAP} = \arg\max_{\tilde{a}} \; p(a_k = \tilde{a}|\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}). \tag{7.38}$$

In this kind of receivers, we see from (7.38) that marginal probabilities $p(a_k|\mathbf{r}, \mathbf{b})$ required by the EM synchronizer may be computed by the symbol detection part of the receiver. Most of the algorithms dedicated to the evaluation of those probabilities may be understood in the context of the factor-graph representation and the SP algorithm: the BCJR algorithm, the turbo decoder and the turbo demodulator presented in chapter 5 are instances of such algorithms. The BCJR algorithm is based on a cycle-free factor graph and can therefore compute exact marginals $p(a_k|\mathbf{r}, \mathbf{b})$ in a finite number of operations. On the contrary, turbo algorithms are based on factor graphs with cycles and are therefore iterative. When the size of the cycles in the graph is large enough, it is usually assumed that the probabilities computed by the SP algorithm are good approximations of actual posterior probabilities $p(a_k|\mathbf{r}, \mathbf{b})$. The E-step can therefore be implemented by applying the SP algorithm on a factor graph which has the general form of the one represented in Fig. 7.2: probabilities $p(a_k|\mathbf{r}, \mathbf{b})$ are equal to the product of the messages entering node a_k (possibly after convergence of the SP algorithm), i.e. $p(y_k|a_k, \mathbf{b})\, p_e(a_k)$. This way of computing marginals $p(a_k|\mathbf{r}, \mathbf{b})$ has been more particularly em-

[4] In the factor-graph framework, both $p(u_l|\mathbf{r}, \mathbf{b})$ and $p(a_k|\mathbf{r}, \mathbf{b})$ are available by applying the SP algorithm on the same factor graph (see chapter 5).

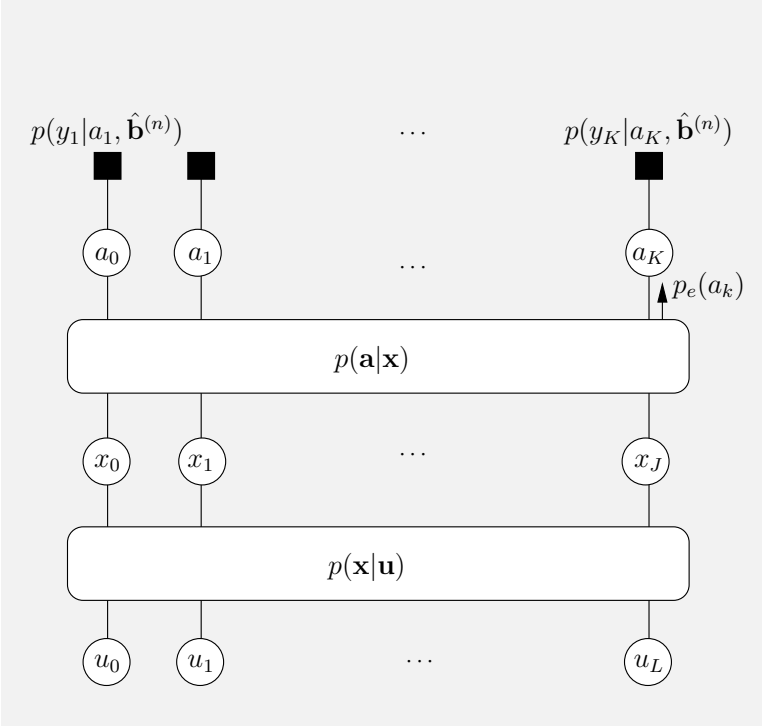

Figure 7.2 Factor-graph for the computation of posterior probabilities $p(a_k|\mathbf{r}, \mathbf{b})$.

phasized in [59, 60] in the context of the EM algorithm. The synchronization algorithm based on this principle is represented in Algorithm 1.

Note that the synchronization algorithm described in Algorithm 1 leads to an important increase of the receiver complexity as compared to conventional FF synchronization methods (see section 7.2). Indeed, on the one hand conventional synchronizers exhibit a low complexity and the receiver overall complexity is therefore dominated[5] by the computation of probabilities $p(a_k|\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}})$. On the other hand, we see from Algorithm 1 that the EM synchronizer implies the computation of $p(a_k|\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}^{(n)})$ at each iteration. In other words, the EM synchronizer implies that the receiver overall complexity is increased by a factor equal to the number of EM iterations performed to synchronize the system.

In order to circumvent this problem, an approximated version of Algorithm 1 have been proposed by Noels et al. in [61]. This modified version of the EM synchronizer reduces the algorithm complexity in the case where the SP algorithm implementing posterior marginal computation is iterative. If the SP algorithm is not iterative, the approach proposed by Noels et al. remains equivalent to Algorithm 1.

Instead of reinitializing all the SP messages and letting the SP algorithm converge at each EM iteration, Noels et. al propose the following approximations (see Algorithm 2):

- The messages in the factor graph are no longer reinitialized at each iteration of the EM algorithm.
- Only one iteration of the SP algorithm is performed at each EM-algorithm iteration. Marginals $p(a_k|\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}^{(n)})$ are approximated by the product of messages entering variable nodes a_k.

Making these approximations, the SP-algorithm iterations are merged with the EM-algorithm iterations. In fact, the resulting algorithm is equivalent to running an SP algorithm for the computation of probabilities $p(a_k|\mathbf{r}, \mathbf{b})$ while updating, at each SP iteration, synchronization estimates according to the EM update rules (7.35)-(7.37). The only extra complexity added by this approach with respect to a receiver synchronized by a FF conventional synchronizer

[5] We assume a CMAP symbol/bit decision rule.

Algorithm 2: Noels et al.'s approximated version of the EM synchronizer

initialization:
  $\hat{\mathbf{b}}^{(0)} = \mathbf{b}_0$;
  `reset(`*factor-graph messages*`);`
repeat
  `compute(`$p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$`);`
    `run(`*1 iteration of the SP algorithm*`);`
    $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)}) \propto p(y_k|a_k, \hat{\mathbf{b}}^{(n)})\, p_e(a_k)$;
  `compute(`$\hat{\mathbf{b}}^{(n+1)} = \arg\max_{\tilde{\mathbf{b}}} \mathcal{Q}(\mathbf{b} = \tilde{\mathbf{b}}, \hat{\mathbf{b}}^{(n)})$`)`
until *convergence*;

is the computation of (7.35)-(7.37) at each SP iteration. The complexity associated with the evaluation of those expressions is usually much smaller than the complexity of one SP iteration. In fact, looking back at example 7.3, we may note that Algorithm 2 is nothing else than a generalization of Lottici&Luise's approach.

Note that, strictly speaking, Algorithm 2 is no longer an EM algorithm. Indeed, performing only one SP iteration at each EM iteration may lead to poor approximations of marginal posterior probabilities $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$, especially during the first EM iterations. Moreover, non-resetting messages in the factor graph implies that the probabilities computed by the SP algorithm will be a function of all the previous estimates $\hat{\mathbf{b}}^{(n)}, \hat{\mathbf{b}}^{(n-1)} \ldots \hat{\mathbf{b}}^{(0)}$ and not only of $\hat{\mathbf{b}}^{(n)}$ as required by the EM algorithm. However, the powerfulness of this approach has already been shown in a number of contributions, see e.g. [62, 63, 64, 65].

7.5 Iterative Synchronization Based on Gradient Methods

In this section, we apply some gradient algorithms, namely the steepest-descent (SD) and the Newton-Raphson (NR) algorithms, to the computation of the CA ML estimate of synchronization parameters.

One of the main criticisms which is usually addressed to the gradient methods is that, unlike the EM algorithm, they imply the computation of the derivatives of the function to maximize (see chapter 6). These derivatives may be difficult to compute, especially if the function to maximize is itself difficult

to evaluate. In the particular case of ML synchronization, for example, the function to maximize may be expressed as

$$\log p(\mathbf{r}|\mathbf{b}) = \log \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{a}), \tag{7.39}$$

i.e. it requires a summation over all the possible transmitted sequences. The evaluation of this function appears therefore quite cumbersome[6] and so might be the evaluation of its derivatives.

In this section, we derive expressions which enable to evaluate the first and second derivatives of objective function $\log p(\mathbf{r}|\mathbf{b})$ with a reasonable complexity. In particular, we show that the evaluation of the first derivative only requires the knowledge of a posteriori marginals $p(a_k|\mathbf{r},\mathbf{b})$. As a consequence, the complexity of one iteration of the SD synchronizer is shown to be similar to one iteration of the EM synchronizer. We then tackle the problem of computing the Hessian matrix required to implement the NR algorithm (see chapter 6). We show that in the general case, the exact evaluation of the Hessian requires the knowledge of the joint posterior probabilities $p(a_k, a_{k'}|\mathbf{r},\mathbf{b})$. We propose a tight approximation of the Hessian matrix in which only the marginals $p(a_k|\mathbf{r},\mathbf{b})$ are required.

7.5.1 The SD algorithm Applied to ML Synchronization

The SD algorithm applied to ML synchronization problem (7.2) generates a sequence of estimates $\hat{\mathbf{b}}^{(n)}$ defined as[7]

$$\hat{\mathbf{b}}^{(n+1)} = \hat{\mathbf{b}}^{(n)} + \alpha^{(n)}\, \nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n)}), \tag{7.40}$$

where $\nabla \log p(\mathbf{r}|\mathbf{b})$ is the gradient of $\log p(\mathbf{r}|\mathbf{b})$ and $\alpha^{(n)}$ defines a well-chosen stepsize factor (see chapter 6). The correction made at each iteration to current estimate $\hat{\mathbf{b}}^{(n)}$ is proportional to the gradient of the objective function. A critical point of the SD algorithm consists therefore in evaluating efficiently the gradient of $\log p(\mathbf{r}|\mathbf{b})$.

In the appendix, we show that $\nabla \log p(\mathbf{r}|\mathbf{b})$ may be expressed as

$$\nabla \log p(\mathbf{r}|\mathbf{b}) = E_{\mathbf{a}|\mathbf{r},\mathbf{b}}\Big[\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\Big], \tag{7.41}$$

[6] We will show in chapter 8 that the evaluation of (7.39) may be tractable in some cases.

[7] We use the shorthand notation $\nabla \log p(\mathbf{r}|\mathbf{b}=\hat{\mathbf{b}}^{(n)}) = \nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n)})$

i.e. it only requires the knowledge of $p(\mathbf{a}|\mathbf{r},\mathbf{b})$ and $\log p(\mathbf{r}|\mathbf{a},\mathbf{b})$. This expression has been proposed in [12] for the efficient evaluation of Cramer-Rao bounds. Here, we will use it for the sake of deriving synchronization algorithms based on the SD method.

Using mathematical model (2.8) and dropping the terms independent of $\mathbf{b}$, we have that

$$\log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \propto \frac{A}{N_0}\,\mathrm{Re}\Big\{\sum_{k=0}^{K-1} a_k^*\, y_k(\tau,\nu)\, e^{-j\theta}\Big\} - \frac{A^2}{2N_0}\sum_{k=0}^{K-1}\|a_k\|^2, \tag{7.42}$$

where $y_k(\tau,\nu)$ is the matched-filter output computed at time $kT+\tau$, assuming frequency offset ν, see (3.27). Plugging (7.42) in (7.41), we get the following close-form expressions of the first derivatives of $\log p(\mathbf{r}|\mathbf{b})$:

$$\frac{\partial}{\partial\tau}\log p(\mathbf{r}|\mathbf{b}) = \frac{A}{N_0}\,\mathrm{Re}\Big\{\sum_{k=0}^{K-1}\eta_k^*(\mathbf{r},\mathbf{b})\frac{\partial y_k(\nu,\tau)}{\partial\tau}\, e^{-j\theta}\Big\}, \tag{7.43}$$

$$\frac{\partial}{\partial\nu}\log p(\mathbf{r}|\mathbf{b}) = \frac{A}{N_0}\,\mathrm{Re}\Big\{\sum_{k=0}^{K-1}\eta_k^*(\mathbf{r},\mathbf{b})\frac{\partial y_k(\nu,\tau)}{\partial\nu}\, e^{-j\theta}\Big\}, \tag{7.44}$$

$$\frac{\partial}{\partial\theta}\log p(\mathbf{r}|\mathbf{b}) = \frac{A}{N_0}\,\mathrm{Im}\Big\{\sum_{k=0}^{K-1}\eta_k^*(\mathbf{r},\mathbf{b})\, y_k(\nu,\tau)\, e^{-j\theta}\Big\}, \tag{7.45}$$

$$\frac{\partial}{\partial A}\log p(\mathbf{r}|\mathbf{b}) = \frac{1}{N_0}\,\mathrm{Re}\Big\{\sum_{k=0}^{K-1}\eta_k^*(\mathbf{r},\mathbf{b})\, y_k(\nu,\tau)\, e^{-j\theta}\Big\} - \frac{A}{N_0}\sum_{k=0}^{K-1}\rho_k(\mathbf{r},\mathbf{b}), \tag{7.46}$$

where $\eta_k(\mathbf{r},\mathbf{b})$ and $\rho_k(\mathbf{r},\mathbf{b})$ have been defined in (7.32)-(7.33) and respectively represent the first and second order a posteriori averages of transmitted symbol a_k.

7.5.2 Implementation of the SD Synchronizer

In this section, we discuss the implementation of the ML SD synchronizer in practical receivers. We see from (7.43)-(7.46) that the evaluation of the

Algorithm 3: Synchronization based on the SD algorithm

initialization:
$\hat{\mathbf{b}}^{(0)} = \mathbf{b}_0$;
repeat
`compute(`$p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$`);`
`reset(`*factor-graph messages*`);`
`run(`*SP algorithm*`);`
$p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)}) \propto p(y_k|a_k, \hat{\mathbf{b}}^{(n)})\, p_e(a_k)$;
`compute(`$\hat{\mathbf{b}}^{(n+1)} = \hat{\mathbf{b}}^{(n)} + \alpha^{(n)} \nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n)})$`);`
until *convergence*;

first derivatives of $\log p(\mathbf{b}|\mathbf{r})$ only requires the knowledge of marginals $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$. The computation of those probabilities clearly dominates the complexity of one SD iteration. As mentioned in section 7.4.3, the evaluation of those probabilities also dominates the complexity of one EM-algorithm iteration. As a consequence, the EM and the SD synchronizers exhibit a similar complexity per iteration.

Note that the discussion in section 7.4.3 about the a posteriori marginal computation may, of course, be applied here. In particular, the symbol decoding part of practical state-of-the-art receivers is well-suited to the computation of marginals $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ since based on CMAP estimation of the transmitted data. The computation of probabilities $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ is usually placed in the context of the factor-graph representation and the SP algorithm: $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ may be computed by applying the SP algorithm on a factor graph having the general form of the one represented in Fig. 7.2. If the factor graph is cycle free, the computation of $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ may be done in a finite number of steps; otherwise, the SP algorithm leads to an iterative algorithm without natural termination. Algorithm 3 summarizes the operations performed by the SD synchronizer when the computation of marginals $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ is implemented by means of the SP algorithm.

In the same way as the EM-based synchronizer, the SD approach defined in Algorithm 3 increases the receiver complexity by a factor equal to the number of SD iterations. Indeed, a receiver synchronized by a conventional FF synchronizer only requires to compute probabilities $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ once, whereas those probabilities have to be recomputed at each iteration of the SD algorithm. When the SP algorithm used to compute marginals $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ is itera-

Algorithm 4: Approximated version of the SD synchronizer

initialization:
  $\hat{\mathbf{b}}^{(0)} = \mathbf{b}_0$;
  `reset(`*factor-graph messages*`);`
repeat
  `compute(`$p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$`);`
    `run(`*1 iteration of the SP algorithm*`);`
    $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)}) \propto p(y_k|a_k, \hat{\mathbf{b}}^{(n)})\, p_e(a_k)$;
  `compute(`$\hat{\mathbf{b}}^{(n+1)} = \hat{\mathbf{b}}^{(n)} + \alpha^{(n)} \nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n)})$`)`
until *convergence*;

tive, the SD algorithm may be simplified by making the same approximations as in section 7.4.3, i.e.

- The factor-graph messages are no longer reinitialized at each iteration of the SD synchronizer.
- Only one iteration of the SP algorithm is performed at each iteration of the SD synchronizer.

Making those approximations, the SD synchronizer reduces to Algorithm 4. The iterations of the SP algorithm are merged with the iterations of the SD algorithm and the receiver complexity is therefore comparable to the complexity of a receiver using conventional synchronization methods. Note, however, that the probabilities used to compute $\nabla \log p(\mathbf{r}|\mathbf{b}^{(n)})$ are no longer equal to actual a posteriori probabilities $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$. Therefore, current estimate $\hat{\mathbf{b}}^{(n)}$ is no longer corrected in the direction of the gradient but rather in the direction of an approximation of it, say $\mathbf{d}^{(n)}$. The convergence of the algorithm may still be guaranteed as long as (see (6.10)):

$$\nabla^T \log p(\mathbf{r}|\mathbf{b}^{(n)})\, \mathbf{d}^{(n)} > 0. \tag{7.47}$$

This condition is however quite difficult to verify in practice.

Another problem associated with the implementation of the SD synchronizer is the choice of the stepsize factor $\alpha^{(n)}$. The value of this factor strongly affects the behavior of the SD algorithm: too small values slow down the convergence whereas too large values lead to divergence of the algorithm. There exist a number of rules for choosing the stepsize $\alpha^{(n)}$ in a gradient descent methods (see [47]). Some of the most widely-used in practice are the minimization rule, the Armijo rule, the Goldstein rule or simply the constant

stepsize. All the stepsize decision rules (but the constant stepsize) imply some evaluations of the objective function. Unfortunately, the evaluation of $\log p(\mathbf{r}|\mathbf{b})$ is not straightforward. Actually, we show in chapter 8 that computing $\log p(\mathbf{r}|\mathbf{b})$ has the same complexity as computing probabilities $p(a_k|\mathbf{r},\mathbf{b})$. The complexity associated with the computation of $\alpha^{(n)}$ is therefore not small and has to be taken into account in the overall complexity of the synchronizer.

7.5.3 The NR Algorithm Applied to ML Synchronization

The NR algorithm generates a sequence of estimates $\hat{\mathbf{b}}^{(n)}$ defined as

$$\hat{\mathbf{b}}^{(n+1)} = \hat{\mathbf{b}}^{(n)} - \alpha^{(n)} \left(\nabla^2 \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n)}) \right)^{-1} \nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n)}). \tag{7.48}$$

We see that, unlike the SD algorithm, the NR approach does no longer correct the current estimate in the direction of the gradient. Instead, it searches a more suitable direction by multiplying the gradient by the inverse of the Hessian matrix. A consequence of this choice is an increase of the speed of convergence. This improvement of the speed of convergence is however at the expense of the recomputation of the Hessian matrix. This operation may be quite complex and efficient expressions have to be derived to evaluate this matrix.

In the appendix, we show that Hessian matrix $\nabla^2 \log p(\mathbf{r}|\mathbf{b})$ may be expressed as

$$\begin{aligned} \nabla^2 \log p(\mathbf{r}|\mathbf{b}) = & E_{\mathbf{a}|\mathbf{r},\mathbf{b}} \left[\nabla^2 \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \right] \\ & + E_{\mathbf{a}|\mathbf{r},\mathbf{b}} \left[\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \, \nabla^T \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \right] \\ & - E_{\mathbf{a}|\mathbf{r},\mathbf{b}} \left[\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \right] \, E_{\mathbf{a}|\mathbf{r},\mathbf{b}} \left[\nabla^T \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \right], \end{aligned} \tag{7.49}$$

i.e. as a function of posterior probability $p(\mathbf{a}|\mathbf{r},\mathbf{b})$ and log-likelihood function $\log p(\mathbf{r}|\mathbf{a},\mathbf{b})$ only.

Let us particularize expression (7.49) to the particular structure of $\log p(\mathbf{r}|\mathbf{a},\mathbf{b})$. For the sake of clarity and conciseness, we will consider $\mathbf{b} = [\tau, \nu, \theta]^T$; the extension to the case $\mathbf{b} = [\tau, \nu, \theta, A]^T$ is straightforward. Taking mathematical

model (2.8) into account and dropping the terms independent of $\mathbf{b}$, we have

$$\begin{aligned}\log p(\mathbf{r}|\mathbf{a},\mathbf{b}) &\propto \frac{A}{N_0}\,\mathrm{Re}\Big\{\sum_{k=0}^{K-1} a_k^*\, y_k(\nu,\tau)\, e^{-j\theta}\Big\},\\ &\propto \mathrm{Re}\Big\{\sum_{k=0}^{K-1} a_k^*\, s_k(\mathbf{b})\Big\},\end{aligned} \tag{7.50}$$

where $s_k(\mathbf{b}) \triangleq \frac{A}{N_0}\, y_k(\nu,\tau)\, e^{-j\theta}$. Plugging expression (7.50) into (7.49), Hessian matrix $\nabla^2 \log p(\mathbf{r}|\mathbf{b})$ may be expressed as

$$\begin{aligned}\nabla^2 \log p(\mathbf{r}|\mathbf{b}) = &\sum_k \mathrm{Re}\Big\{\eta_k^*(\mathbf{r},\mathbf{b})\ \nabla^2 s_k(\mathbf{b})\Big\}\\ &+\sum_{k,k'} \mathrm{Re}\Big\{\kappa_{k,k'}(\mathbf{r},\mathbf{b})\ \nabla s_k(\mathbf{b})\nabla^T s_{k'}^*(\mathbf{b})\Big\}\\ &+\sum_{k,k'} \mathrm{Re}\Big\{\varepsilon_{k,k'}^*(\mathbf{r},\mathbf{b})\nabla s_k(\mathbf{b})\nabla^T s_{k'}(\mathbf{b})\Big\},\end{aligned} \tag{7.51}$$

where $\eta_k(\mathbf{r},\mathbf{b})$ has been defined in (7.32) whereas $\kappa_{k,k'}(\mathbf{r},\mathbf{b})$ and $\varepsilon_{k,k'}(\mathbf{r},\mathbf{b})$ are defined as follows

$$\kappa_{k,k'}(\mathbf{r},\mathbf{b}) \triangleq \sum_{a',a\in\mathcal{A}} a^*\, a'\, p(a_k = a, a_{k'} = a'|\mathbf{r},\mathbf{b}) - \eta_k^*(\mathbf{r},\mathbf{b})\,\eta_{k'}(\mathbf{r},\mathbf{b}), \tag{7.52}$$

$$\varepsilon_{k,k'}(\mathbf{r},\mathbf{b}) \triangleq \sum_{a,a'\in\mathcal{A}} a\, a'\ p(a_k = a, a_{k'} = a'|\mathbf{r},\mathbf{b}) - \eta_k(\mathbf{r},\mathbf{b})\,\eta_{k'}(\mathbf{r},\mathbf{b}). \tag{7.53}$$

7.5.4 Implementation of the NR Synchronizer

From definitions (7.52) and (7.53), we see that, unlike the EM or the SD synchronizers, the implementation of the NR maximization method requires the knowledge of joint probabilities $p(a_k, a_{k'}|\mathbf{r},\mathbf{b})$. The computation of these probabilities may be implemented via an SP algorithm applied on the proper factor graph[8]. The implementation of the NR synchronizer based on this approach is represented in Algorithm 5.

From (7.51), we note that the second and third terms of the Hessian matrix require a double summation over the transmitted symbols. These terms contain therefore a large number of terms, making the evaluation of (7.51) quite complex. We therefore consider an approximated version of the NR synchronizer in which we assume that

$$p(a_k, a_{k'}|\mathbf{r},\mathbf{b}) \simeq p(a_k|\mathbf{r},\mathbf{b})\, p(a_{k'}|\mathbf{r},\mathbf{b}), \tag{7.54}$$

[8]This factor graph has not been represented here.

Algorithm 5: Synchronization based on the NR algorithm

initialization:

$\hat{\mathbf{b}}^{(0)} = \mathbf{b}_0$;

repeat

`compute(`$p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ *and* $p(a_k, a_{k'}|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$`);`

`reset(`*factor-graph messages*`);`

`run(`*SP algorithm*`);`

$p(a_k, a_{k'}|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ = *product of messages entering node* $\mathbf{a}_{k,k'} = [a_k, a_{k'}]$;

$p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)}) = \sum_{a_{k'}} p(a_k, a_{k'}|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$;

`compute(`$\hat{\mathbf{b}}^{(n+1)} = \hat{\mathbf{b}}^{(n)} - \alpha^{(n)} \left(\nabla^2 \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n)})\right)^{-1} \nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n)})$`);`

until *convergence*;

i.e. joint probability $p(a_k, a_{k'}|\mathbf{r}, \mathbf{b})$ is approximated by the product of its marginals. Making this approximation, we may note that $\kappa_{k,k'}(\mathbf{r}, \mathbf{b}) = 0$ and $\varepsilon_{k,k'}(\mathbf{r}, \mathbf{b}) = 0 \ \forall \ k \neq k'$. As a consequence, (7.50) reduces to

$$\begin{aligned}\nabla^2 \log p(\mathbf{r}|\mathbf{b}) = &\sum_k \mathrm{Re}\Big\{\eta_k^*(\mathbf{r}, \mathbf{b}) \ \nabla^2 s_k(\mathbf{b})\Big\} \\ &+ \sum_k \mathrm{Re}\Big\{\kappa_{k,k}(\mathbf{r}, \mathbf{b}) \ \nabla s_k(\mathbf{b}) \nabla^T s_k^*(\mathbf{b})\Big\} \\ &+ \sum_k \mathrm{Re}\Big\{\varepsilon_{k,k}^*(\mathbf{r}, \mathbf{b}) \nabla s_k(\mathbf{b}) \nabla^T s_k(\mathbf{b})\Big\}. \end{aligned} \tag{7.55}$$

The double summation is therefore avoided. Note that we can come to the same conclusion without making approximation (7.54) but rather using the law of large numbers [66]. Indeed, summation $\sum_k \sum_{k' \neq k} \mathrm{Re}\Big\{\kappa_{k,k'}(\mathbf{r}, \mathbf{b}) \ \nabla s_k(\mathbf{b}) \nabla^T s_{k'}^*(\mathbf{b})\Big\}$ contains a large number of terms and the mean of each of them is equal to zero. Provided that the number of terms is large enough, we have therefore

$$\sum_k \sum_{k' \neq k} \mathrm{Re}\Big\{\kappa_{k,k'}(\mathbf{r}, \mathbf{b}) \ \nabla s_k(\mathbf{b}) \nabla^T s_{k'}^*(\mathbf{b})\Big\} \simeq 0. \tag{7.56}$$

The same reasoning is valid for $\sum_k \sum_{k' \neq k} \mathrm{Re}\Big\{\varepsilon_{k,k'}^*(\mathbf{r}, \mathbf{b}) \ \nabla s_k(\mathbf{b}) \nabla^T s_{k'}(\mathbf{b})\Big\}$ and using these results we get therefore (7.55).

Approximation (7.54) may be combined with Noels et al's approximation for the computation of marginals $p(a_k|\mathbf{r}, \mathbf{b})$ (see section 7.4.3). Based on these two approximations, we come up with the synchronization algorithm defined in Algorithm 6.

Algorithm 6: Approximated version of the NR synchronizer

initialization:

$\hat{\mathbf{b}}^{(0)} = \mathbf{b}_0$;

`reset(`*factor-graph messages*`);`

repeat

`compute(`$p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ *and* $p(a_k, a_{k'}|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$`);`

`run(`*1 iteration of the SP algorithm*`);`

$p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)}) \simeq p(y_k|a_k, \hat{\mathbf{b}}^{(n)})\, p_e(a_k)$;

$p(a_k, a_{k'}|\mathbf{r}, \hat{\mathbf{b}}^{(n)}) \simeq p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})\, p(a_{k'}|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$;

`compute(`$\hat{\mathbf{b}}^{(n+1)} = \hat{\mathbf{b}}^{(n)} - \alpha^{(n)} \left(\nabla^2 \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n)})\right)^{-1} \nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n)})$`);`

until *convergence*;

As mentioned in chapter 6, another problem associated to the implementation of an NR algorithm is that it can converge to a minimum as well as to a maximum of the objective function. This drawback can be avoided by considering the diagonal elements of the Hessian matrix and the particular structure of the likelihood function in synchronization problems. Indeed, convergence to a minimum or a saddle point can be detected by considering the diagonal elements of the Hessian matrix: if a given element is positive, it means that the corresponding synchronization parameter is updated in the direction of a minimum. If the convergence to a minimum or a saddle point is detected, the synchronization-parameter estimate may be corrected by taking into account the periodic structure of the log-likelihood function: the maxima, minima and saddle points of $p(\mathbf{r}|\mathbf{b})$ are separated by a known distance. For example, in a carrier phase estimation problem for BPSK transmission, we know that the maxima and the minima of the likelihood function are usually separated by a distance $\pi/2$. Therefore, if the convergence to a minimum or a saddle point is detected, the estimate can be corrected by simply subtracting this known distance.

When the number of synchronization parameters becomes large, the inversion of the Hessian $\nabla^2 \log p(\mathbf{r}|\mathbf{b})$ may exhibit a too high complexity. In order to circumvent this problem a "sequential" version of the NR synchronizer may be considered: instead of updating the estimate over the overall synchronization parameter space $\mathcal{B}$, we can sequentially update the estimate over smaller subspaces $\mathcal{B}_i$ (with $\bigcup_i \mathcal{B}_i = \mathcal{B}$). Indeed, as long as each successive updated estimate increases the objective function $\log p(\mathbf{r}|\mathbf{b})$, we are ensured

by the Global Convergence Theorem (see section 6.1.1) to converge to a maximum (or a saddle point) of $\log p(\mathbf{r}|\mathbf{b})$. Since the dimension of each subspace is smaller than the dimension of $\mathcal{B}$, the application of the NR algorithm on each subspace will also lead to the inversion of Hessian matrices of smaller dimension. Note, however, that this simplification is at the expense of the recomputation of the symbol a posteriori marginals $p(a_k|\mathbf{r}, \mathbf{b})$ for each subspace.

7.6 Combining the Gradient and the EM Synchronization Frameworks

Both the EM and the gradient synchronization frameworks have some advantages and some drawbacks. On the one hand, the EM algorithm is quite robust and its associated update equations (see (7.35)-(7.37)) rather simple to implement. Unfortunately, its convergence is sometimes dramatically slow. On the other hand, the NR algorithm exhibits a high speed of convergence but its robustness is often more critical. Depending on the considered case (or synchronization parameter to estimate), we may therefore prefer one framework to the other one.

Let us assume that the EM synchronization framework would be well-suited to a subspace $\mathcal{B}_1 \subset \mathcal{B}$ whereas we would prefer to apply the gradient algorithm on a subspace $\mathcal{B}_2 \subset \mathcal{B}$ (with $\mathcal{B}_1 \cup \mathcal{B}_2 = \mathcal{B}$). By the Global Convergence theorem (see section 6.1.1), it is clear that the convergence to a stationary point of the likelihood function is still ensured by applying the EM algorithm on subspace $\mathcal{B}_1$ and the gradient algorithm on subspace $\mathcal{B}_2$. Indeed, both algorithms ensure[9] an increase of the likelihood function as long as a stationary point has not been reached. Since $\mathcal{B}_1$ and $\mathcal{B}_2$ form a partition of $\mathcal{B}$, the combination of the EM and gradient algorithms will still ensure the convergence to a stationary point of the likelihood function.

As a consequence, if the EM algorithm exhibits a very slow speed of convergence on some parameter subspaces, we can instead use a NR algorithm to increase the speed of convergence. On the contrary, if the EM algorithm converges rapidly for some parameters, its use can be preferred to a NR algorithm since it is more robust and (slightly) less complex to implement. Note, however, that the application of a different synchronization framework on each

[9] provided that some conditions are verified (see chapter 6).

subspace requires to recompute symbol a posteriori marginals $p(a_k|\mathbf{r},\mathbf{b})$ for each subspace.

7.7 The SP Algorithm: a General Framework for Iterative Synchronization

The frameworks based on the EM algorithm (see section 7.4) and the gradient algorithms (see section 7.5) give smart solutions for CA SDD synchronization when marginal a posteriori probabilities can be exactly computed. When these frameworks are applied to turbo receivers, these marginal probabilities are often approximated (see sections 7.4.3, 7.5.2 and 7.5.4) and we can therefore ask the question of the validity of these approximations.

In this section, we place SDD iterative synchronization into the context of the factor-graph representation and the SP algorithm (see chapter 5). This approach will allow us to answer the question of which soft information should be used by the SDD synchronizers. In particular, it will give us a new insight into the approximations made on the a posteriori marginal probabilities in sections 7.4.3, 7.5.2 and 7.5.4.

This section is organized as follows. We first derive the general expressions of the SP algorithm associated to the bit-wise optimal receiver (4.4). We then consider a suboptimal version of the SP algorithm by approximating some of the messages by a delta function. This modification of the SP algorithm defines a new family of SDD iterative synchronizers whose feature is to compute new synchronization estimates by considering the symbol extrinsic probabilities as symbol a priori information.

Based on this principle, different iterative synchronizers are proposed. In particular, we apply the frameworks for ML estimation derived in sections 7.4 and 7.5 to the resolution of a modified ML problem arising in the SP-algorithm framework. Doing so, we show that the modified version of the EM (resp. SD and NR) synchronizer defined in Algorithm 2 (resp. 4 and 6) is a particular case of a more general synchronization framework based on both the SP and the EM (resp. SD and NR) algorithms.

7.7.1 The SP-algorithm Applied to Bit-wise Optimal Reception

In order to better understand the synchronization algorithms which will be derived in the sequel of this section, let us take a step back and consider the design of a bit-wise optimal receiver in the presence of synchronization parameter uncertainties.

From (4.4), we have that the implementation of the bit-wise optimal receiver requires the computation of marginals $p(u_l|\mathbf{r})$. These marginals may be expressed as

$$p(u_l|\mathbf{r}) \propto \sum_{\sim\{u_l\}} p(\mathbf{r}, \mathbf{a}, \mathbf{x}, \mathbf{u}, \mathbf{b}). \tag{7.57}$$

Probabilities $p(u_l|\mathbf{r})$ may be (exactly or approximately)[10] computed by applying the SP algorithm on the factor-graph representation of $p(\mathbf{r}, \mathbf{a}, \mathbf{x}, \mathbf{u}, \mathbf{b})$. Using Bayes rule and the (conditional) independence between some of the random variables, we have that $p(\mathbf{r}, \mathbf{a}, \mathbf{x}, \mathbf{u}, \mathbf{b})$ may be factorized as

$$\begin{aligned} p(\mathbf{r}, \mathbf{a}, \mathbf{x}, \mathbf{u}, \mathbf{b}) &= p(\mathbf{r}|\mathbf{a}, \mathbf{x}, \mathbf{u}, \mathbf{b})\, p(\mathbf{a}, \mathbf{x}, \mathbf{u}, \mathbf{b}) \\ &= p(\mathbf{r}|\mathbf{a}, \mathbf{b})\, p(\mathbf{a}|\mathbf{x}, \mathbf{u}, \mathbf{b})\, p(\mathbf{x}, \mathbf{u}, \mathbf{b}) \\ &= p(\mathbf{r}|\mathbf{a}, \mathbf{b})\, p(\mathbf{a}|\mathbf{x})\, p(\mathbf{x}|\mathbf{u})\, p(\mathbf{u})\, p(\mathbf{b}). \end{aligned} \tag{7.58}$$

Factor $p(\mathbf{r}|\mathbf{a}, \mathbf{b})$ accounts for the effect of the channel whereas $p(\mathbf{a}|\mathbf{x})$ and $p(\mathbf{x}|\mathbf{u})$ respectively model the behavior of the bit-to-symbol mapping and the error-correcting code used for the transmission. $p(\mathbf{u})$ and $p(\mathbf{b})$ are the a priori information about the transmitted sequence and the synchronization parameters, respectively. Both distributions will be assumed uniform in the sequel.

The factor-graph representation associated to (7.58) is represented in Fig. 7.3. Probability $p(\mathbf{a}|\mathbf{x})$ depends on the bit-to-symbol mapping and may be factorized as in (5.41). Factor $p(\mathbf{x}|\mathbf{u})$ is a function of the error-correcting code which is used for the transmission: if a convolutional code is used, $p(\mathbf{x}|\mathbf{u})$ factors as in (5.16); for a turbo-coded transmission, factorization (5.33) has to be considered.

As long as the code used for the transmission is not trivial, the dependence between the coded bits introduces cycles in the graph represented in Fig. 7.3.

[10] depending on whether the graph contains cycles or not (see chapter 3).

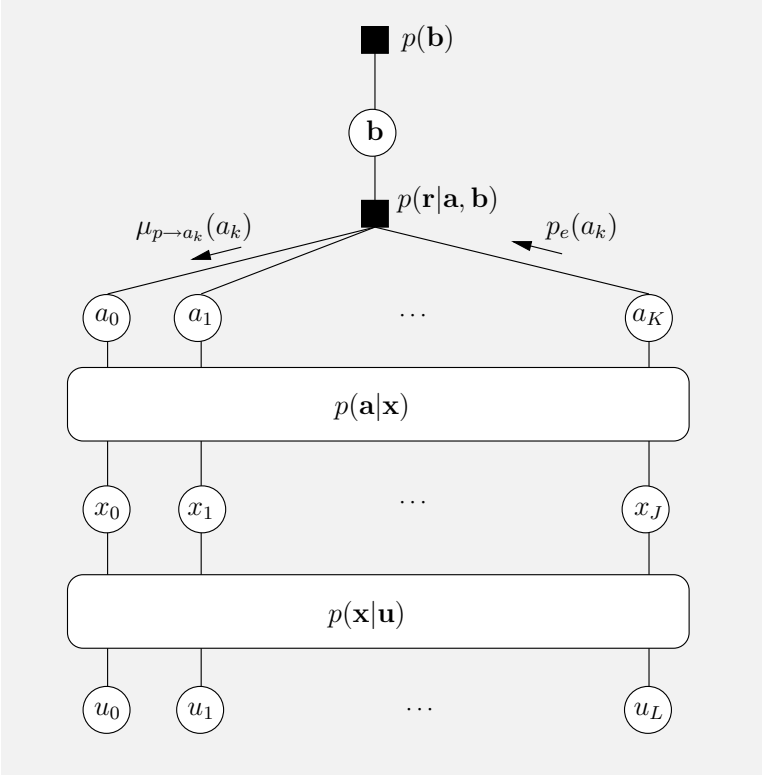

Figure 7.3 Factor-graph representation of $p(\mathbf{r},\mathbf{a},\mathbf{x},\mathbf{u},\mathbf{b})$.

A consequence of the presence of cycles is that the application of the SP algorithm leads to an iterative algorithm with no natural termination (see chapter 5). It is therefore required to define a message-passing schedule in order to specify the messages which are updated at each step of the algorithm. Using the notation of Fig. 7.3, at each iteration we first update messages $\mu_{p\to a_k}(a_k)$, i.e.

$$\begin{aligned}\mu_{p\to a_k}(a_k) &= \sum_{\sim\{a_k\}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{b}) \prod_{k'\neq k} p_e(a_{k'}),\\ &= \sum_{\mathbf{a}\setminus\{a_k\}} \int_{\mathbf{b}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{b}) \prod_{k'\neq k} p_e(a_{k'})\, d\mathbf{b}. \end{aligned} \tag{7.59}$$

Then, extrinsic probabilities $p_e(a_k)$ (see chapter 5) are recomputed by applying the SP algorithm on the code and mapping factor graphs. Note that, since the factor graphs of $p(\mathbf{a}|\mathbf{x})$ and $p(\mathbf{x}|\mathbf{u})$ remain the same, the message update equations derived in examples 5.4-5.6 for the computation of $p_e(a_k)$ are still

valid here.

Another consequence of the presence of cycles in the factor graph is that it can no longer be proved that the results delivered by the SP algorithm are exact. Provided that the length of the cycles is large enough, it seems however that the probabilities delivered by the SP algorithm are, after convergence, good approximations of the actual marginals. Denoting by $p^{(n)}(u_l|\mathbf{r})$ the approximation of the actual marginal $p(u_l|\mathbf{r})$ at iteration n, we have therefore

$$p(u_l|\mathbf{r}) \simeq \lim_{n\to\infty} p^{(n)}(u_l|\mathbf{r}). \tag{7.60}$$

7.7.2 Message Approximations and Iterative Synchronization

The application of the SP algorithm for the computation of marginals $p(u_l|\mathbf{r})$ requires to compute an integral over the synchronization-parameter space in (7.59). This integral does not have any analytical solution and alternative approaches have to be considered to evaluate (7.59). In this section, we propose some approximations which simplify the implementation of the SP algorithm. These approximations will lead to a synchronization framework based on the SP algorithm.

Let us define[11] $\forall\ a_k$,

$$p_k^{(n)}(a_k|\mathbf{r},\mathbf{b}) \propto \sum_{\mathbf{a}\setminus\{a_k\}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{b}) \prod_{k'\neq k} p_e^{(n)}(a_{k'}), \tag{7.61}$$

$$p_k^{(n)}(\mathbf{b}|\mathbf{r}) \propto \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{b}) \prod_{k'\neq k} p_e^{(n)}(a_{k'}). \tag{7.62}$$

Using these definitions, message $\mu_{p\to a_k}^{(n)}(a_k)$ in (7.59) may be rewritten as

$$\mu_{p\to a_k}^{(n)}(a_k) = \int_{\mathbf{b}} p_k^{(n)}(a_k|\mathbf{r},\mathbf{b})\, p_k^{(n)}(\mathbf{b}|\mathbf{r})\, d\mathbf{b}. \tag{7.63}$$

Based on the latter expression, we consider the following approximations:

- We assume that any message $p_k^{(n)}(\mathbf{b}|\mathbf{r})$ may be well-approximated as[12]

$$p_k^{(n)}(\mathbf{b}|\mathbf{r}) \simeq p^{(n)}(\mathbf{b}|\mathbf{r}), \tag{7.64}$$

[11] Notation $p^{(n)}(\cdot)$ (or $\mu^{(n)}(\cdot)$) denotes that the message is computed at the n^{th} iteration of the SP algorithm.

[12] This approximation has also been proposed in [67].

where

$$p^{(n)}(\mathbf{b}|\mathbf{r}) \triangleq \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b}) \prod_k p_e^{(n)}(a_k). \tag{7.65}$$

In other words, we assume that taking all the extrinsic probabilities $p_e(a_k)$ into account in the computation of probabilities $p_k^{(n)}(\mathbf{b}|\mathbf{r})$ will not significantly modify the result. Indeed, any symbol a_k only (significantly) affects a limited number of observation samples in $\mathbf{r}$. Therefore, modifying the a priori probability of one symbol when the size of $\mathbf{r}$ is large, should not lead to significant variations in $p_k^{(n)}(\mathbf{b}|\mathbf{r})$.

Note that doing this approximation, all messages $p_k^{(n)}(\mathbf{b}|\mathbf{r})$ are equal to $p^{(n)}(\mathbf{b}|\mathbf{r})$ and have therefore to be computed only once.

- We assume that probability $p^{(n)}(\mathbf{b}|\mathbf{r})$ may be well-approximated by a delta function[13], i.e.

$$p^{(n)}(\mathbf{b}|\mathbf{r}) \simeq \delta(\mathbf{b} - \hat{\mathbf{b}}^{(n)}). \tag{7.66}$$

In fact, this approximation makes sense in the context of synchronization. Indeed, the same approximation, see (4.8), on actual marginal probability $p(\mathbf{b}|\mathbf{r})$ leads to the synchronization approach introduced in chapter 4. Now, $p^{(n)}(\mathbf{b}|\mathbf{r})$ is actually an approximation of $p(\mathbf{b}|\mathbf{r})$. Indeed, $p(\mathbf{b}|\mathbf{r})$ may be viewed as the marginal of joint probability $p(\mathbf{r},\mathbf{a},\mathbf{x},\mathbf{u},\mathbf{b})$, i.e.

$$p(\mathbf{b}|\mathbf{r}) \propto \sum_{\sim\{\mathbf{b}\}} p(\mathbf{r},\mathbf{a},\mathbf{x},\mathbf{u},\mathbf{b}). \tag{7.67}$$

Comparing (7.67) with (7.57), we see that marginals $p(u_k|\mathbf{r})$ and $p(\mathbf{b}|\mathbf{r})$ may be computed by applying the SP algorithm on the same factor graph. Therefore, the product of messages entering synchronization node $\mathbf{b}$ in Fig. 7.3 is an approximation of actual marginal $p(\mathbf{b}|\mathbf{r})$. Now, applying the SP-algorithm update rules, we have that the product of the messages entering node $\mathbf{b}$ is equal to

$$p^{(n)}(\mathbf{b}|\mathbf{r}) = \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{b}) \prod_k p_e^{(n)}(a_k). \tag{7.68}$$

In this context, if (4.8) is a tight approximation, we may expect (7.66) to be also valid.

[13] The idea of approximating SP messages of continuous variables by canonical distributions has first been generalized by Worthen in [68]. More complicated approximations than a delta function may be considered, see e.g. [20, 67, 69].

Algorithm 7: Synchronization based on the SP algorithm

initialization:
  $\hat{\mathbf{b}}^{(0)} = \mathbf{b}_0$;
  `reset(`*factor-graph messages*`);`
repeat
  `compute(`$p_e^{(n)}(a_k)$`);`
  `compute(`$\hat{\mathbf{b}}^{(n)} \sim$ *"well-chosen" point of* $p^{(n)}(\mathbf{b}|\mathbf{r})$`)`
until *convergence*;

Approximation (7.64) and (7.66) enable to simplify the implementation of the SP algorithm. Indeed, using these approximations, messages $\mu_{p\to a_k}^{(n)}(a_k)$ may simply be computed as

$$\begin{aligned}\mu_{p\to a_k}^{(n)}(a_k) &= \int_{\mathbf{b}} p_k^{(n)}(a_k|\mathbf{r},\mathbf{b})\,\delta(\mathbf{b}-\hat{\mathbf{b}}^{(n)})\,d\mathbf{b},\\ &= p_k^{(n)}(a_k|\mathbf{r},\mathbf{b}=\hat{\mathbf{b}}^{(n)}),\\ &= p_k^{(n)}(a_k|\mathbf{y},\mathbf{b}=\hat{\mathbf{b}}^{(n)}),\end{aligned} \tag{7.69}$$

where the last equality follows from the fact that $\mathbf{y}$ is a sufficient statistic for symbol estimation (see Section 3.6). Using definition (7.61) and the fact that the noise which affects the observations is white, we have

$$\begin{aligned}\mu_{p\to a_k}^{(n)}(a_k) &\propto \sum_{\mathbf{a}\setminus\{a_k\}} p(\mathbf{y}|\mathbf{a},\mathbf{b}=\hat{\mathbf{b}}^{(n)}) \prod_{k'\neq k} p_e^{(n)}(a_{k'}),\\ &\propto \sum_{\mathbf{a}\setminus\{a_k\}} \prod_{k'} p(y_{k'}|a_{k'},\mathbf{b}=\hat{\mathbf{b}}^{(n)}) \prod_{k'\neq k} p_e^{(n)}(a_{k'}),\\ &\propto p(y_k|a_k,\mathbf{b}=\hat{\mathbf{b}}^{(n)}) \prod_{k'\neq k}\sum_{a_{k'}} p(y_{k'}|a_{k'},\mathbf{b})\,p_e^{(n)}(a_{k'}).\end{aligned} \tag{7.70}$$

Note that the last factor in the right-hand term of (7.70) does not depend on a_k and we have therefore

$$\mu_{p\to a_k}^{(n)}(a_k) \propto p(y_k|\,a_k,\mathbf{b}=\hat{\mathbf{b}}^{(n)}). \tag{7.71}$$

The evaluation of $\mu_{p\to a_k}^{(n)}(a_k)$ is easy since $p(y_k|\,a_k,\mathbf{b}=\hat{\mathbf{b}}^{(n)})$ is a Gaussian distribution.

We therefore come up with a two-step iterative synchronization procedure which may be described as follows. In a first step, the extrinsic probabilities $p_e^{(n)}(a_k)$ are computed by assuming that the synchronization parameters are perfectly known and equal to the estimate computed at the previous iteration,

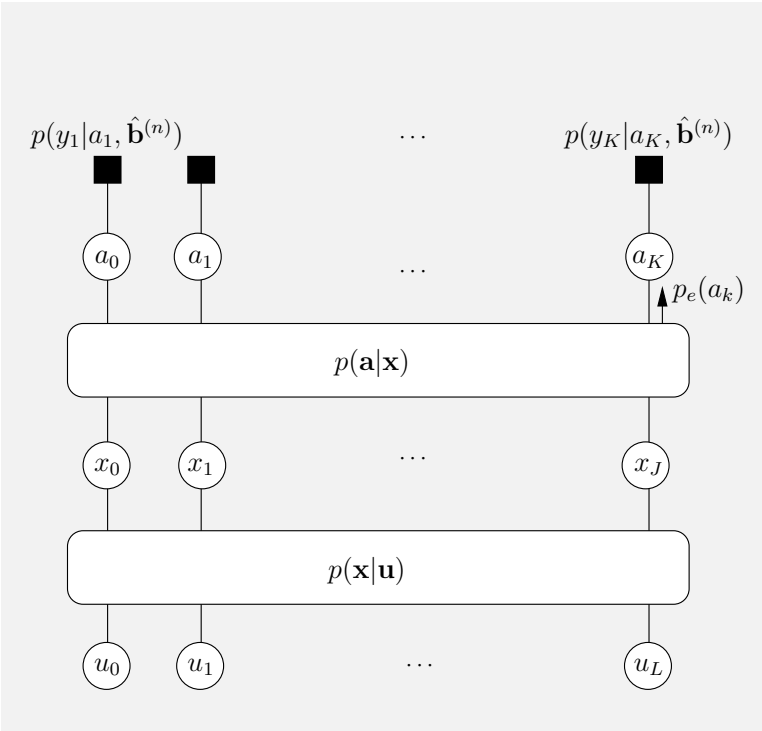

Figure 7.4 Simplified factor-graph for the computation of extrinsic probabilities $p_e(a_k)$.

i.e. $\hat{\mathbf{b}}^{(n-1)}$. The extrinsic probabilities can thus be computed by applying the SP algorithm to the simplified factor graph represented in Fig. 7.4. Then, in a second step, a new estimate $\hat{\mathbf{b}}^{(n)}$ is computed from distribution $p^{(n)}(\mathbf{b}|\mathbf{r})$. This algorithm is summarized in Algorithm 7.

7.7.3 ML Update of the SP Estimates

From Algorithm 7, we see that $\hat{\mathbf{b}}^{(n)}$ has to be computed by choosing a proper point of distribution $p^{(n)}(\mathbf{b}|\mathbf{r})$. In this work, we propose to compute the new estimate $\hat{\mathbf{b}}^{(n)}$ by maximizing $p^{(n)}(\mathbf{b}|\mathbf{r})$, i.e.

$$\hat{\mathbf{b}}^{(n)} = \arg\max_{\tilde{\mathbf{b}}} \log p^{(n)}(\mathbf{b} = \tilde{\mathbf{b}}|\mathbf{r}). \tag{7.72}$$

Note that we usually do not have any a priori information about the synchronization parameters and $p(\mathbf{b})$ is therefore uniform. In that case,

$$p^{(n)}(\mathbf{b}|\mathbf{r}) \propto p^{(n)}(\mathbf{r}|\mathbf{b}). \tag{7.73}$$

The computation of $\hat{\mathbf{b}}^{(n)}$ may then be based on $p^{(n)}(\mathbf{r}|\mathbf{b})$ instead of $p^{(n)}(\mathbf{b}|\mathbf{r})$. In particular, maximization problem (7.72) reduces to an ML estimation problem:

$$\hat{\mathbf{b}}^{(n)} = \arg\max_{\tilde{\mathbf{b}}} \log p^{(n)}(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}}). \tag{7.74}$$

From (7.74), we see that we have to solve, at each iteration of the SP synchronizer, a modified ML problem in which extrinsic probabilities $p_e(a_k)$ are used as a priori information on the transmitted symbols. This approach is actually equivalent to the approach proposed by Zhang and Burr [58] in an ad-hoc way to derive their SDD phase synchronizer (see example 7.4). Zhang&Burr's phase synchronizer may therefore be regarded as an approximation of the SP algorithm.

The maximization of $p^{(n)}(\mathbf{r}|\mathbf{b})$ is not a trivial problem since no analytical solutions exist. Note, however, that maximization problem (7.74) is nothing else than a maximum-likelihood problem in which the actual sequence a priori probability $p(\mathbf{a})$ has been replaced by $p_e^{(n)}(\mathbf{a}) \triangleq \prod_k p_e^{(n)}(a_k)$. Therefore, the frameworks based on the EM algorithm and the gradient algorithm remain valid for the maximization of $p^{(n)}(\mathbf{r}|\mathbf{b})$, by simply replacing $p(\mathbf{a})$ by $p_e^{(n)}(\mathbf{a})$. In the remainder of this section, we will present iterative synchronization methods obtained by combining the SP synchronizer with the frameworks derived in sections 7.4 and 7.5.

7.7.4 Fixed Points of the SP Synchronizer

In this section, we consider the fixed points of the SP synchronizer when $\hat{\mathbf{b}}^{(n)}$ is computed at each iteration according to an ML criterion (7.74). We show that when the factor graph on which is based the computation of the extrinsic probabilities $p_e^{(n)}(\mathbf{a})$ (see Fig. (7.4)) is cycle-free, the fixed points of the SP synchronizer have to be the stationary points of $\log p(\mathbf{r}|\mathbf{b})$.

Let us first notice that, if an estimate $\hat{\mathbf{b}}^{(n-1)}$ is a fixed point of the SP synchronizer (7.74), it has to cancel the gradient of the modified likelihood func-

tion $\log p^{(n)}(\mathbf{r}|\mathbf{b})$. Indeed, let us assume that

$$\nabla \log p^{(n)}(\mathbf{r}|\hat{\mathbf{b}}^{(n-1)}) \neq 0. \tag{7.75}$$

Then, $\hat{\mathbf{b}}^{(n-1)}$ is not a maximum of $\log p^{(n)}(\mathbf{r}|\mathbf{b})$ and we have from (7.74) that $\hat{\mathbf{b}}^{(n)} \neq \hat{\mathbf{b}}^{(n-1)}$. By contraposition, we have therefore

$$\nabla \log p^{(n)}(\mathbf{r}|\hat{\mathbf{b}}^{(n-1)}) = 0, \tag{7.76}$$

as long as $\hat{\mathbf{b}}^{(n-1)}$ is a fixed point of the SP synchronizer. Now, it is shown in the appendix that

$$\nabla \log p^{(n)}(\mathbf{r}|\hat{\mathbf{b}}^{(n-1)}) = \nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n-1)}), \tag{7.77}$$

as long as the factor graph used to compute the extrinsic probabilities $p_e^{(n)}(a_k)$ is cycle free. In other words, the gradient of the modified likelihood function $\log p^{(n)}(\mathbf{r}|\mathbf{b})$ and the gradient of the actual likelihood function $\log p(\mathbf{r}|\mathbf{b})$ are equal when evaluated at the previous estimate, i.e. when $\mathbf{b} = \hat{\mathbf{b}}^{(n-1)}$. As a consequence, if $\hat{\mathbf{b}}^{(n-1)}$ cancels the gradient of the modified likelihood function, it also cancels the gradient of the actual likelihood function:

$$\nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n-1)}) = 0. \tag{7.78}$$

It is clear from (7.78) that any fixed point of the SP synchronizer is also a stationary point of $\log p(\mathbf{r}|\mathbf{b})$.

Note, however, that we cannot prove that the mapping defined in (7.74) is such that

$$\log p(\mathbf{r}|\mathbf{b} = \hat{\mathbf{b}}^{(n)}) > \log p(\mathbf{r}|\mathbf{b} = \hat{\mathbf{b}}^{(n-1)}), \tag{7.79}$$

as long as a stationary point of $\log p(\mathbf{r}|\mathbf{b})$ has not been reached. Therefore, the convergence of the SP synchronizer to these stationary points can not be proved by the Global Convergence Theorem (see section 6.1.1). Nevertheless, if the sequence defined by the SP synchronizer converges, we are ensured that the convergence point is a stationary point of the actual likelihood function $\log p(\mathbf{r}|\mathbf{b})$.

7.7.5 Combining the SP Algorithm and the EM Algorithm

In this section, we apply the EM algorithm to the following maximization problem:

$$\hat{\mathbf{b}}^{(n)} = \arg\max_{\tilde{\mathbf{b}}} \log p^{(n)}(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}}), \tag{7.80}$$

where

$$p^{(n)}(\mathbf{r}|\mathbf{b}) = \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p_e^{(n)}(\mathbf{a}). \tag{7.81}$$

Comparing (7.81) with (7.3), we see that the modified likelihood function $p^{(n)}(\mathbf{r}|\mathbf{b})$ has exactly the same structure as the actual likelihood function $p(\mathbf{r}|\mathbf{b})$ but it uses $p_e^{(n)}(\mathbf{a}) = \prod_k p_e^{(n)}(a_k)$ instead of $p(\mathbf{a})$ as sequence a priori information. We will show that the particular factorization of $p_e^{(n)}(\mathbf{a})$ simplifies the implementation of the EM algorithm.

Defining the complete data set as $\mathbf{z} = [\mathbf{r}^T, \mathbf{a}^T]^T$ and following the same reasoning as in section 7.4.3, we have that the function which has to be maximized at each EM iteration is given by

$$\mathcal{Q}^{(n)}(\mathbf{b}, \hat{\mathbf{b}}^{(n,m)}) = \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}^{(n,m)}) \log p(\mathbf{r}|\mathbf{a},\mathbf{b}), \tag{7.82}$$

where $\hat{\mathbf{b}}^{(n,m)}$ denotes the m^{th} estimate generated by the EM algorithm at the n^{th} iteration of the SP algorithm and $p^{(n)}(\mathbf{a}|\mathbf{r},\mathbf{b})$ may be written as

$$p^{(n)}(\mathbf{a}|\mathbf{r},\mathbf{b}) \propto p(\mathbf{r}|\mathbf{a},\mathbf{b}) p_e^{(n)}(\mathbf{a}). \tag{7.83}$$

Since (7.82) has the same structure as (7.30), expressions (7.35)-(7.37) still provide the solution of the M-step. However, due to the particular form of $p^{(n)}(\mathbf{a}|\mathbf{r},\mathbf{b})$, the computation of posterior average values $\eta(\mathbf{r}, \hat{\mathbf{b}}^{(n,m)})$ and $\rho(\mathbf{r}, \hat{\mathbf{b}}^{(n,m)})$ is simplified. Indeed we have by definition that

$$\begin{aligned}
\eta(\mathbf{r}, \hat{\mathbf{b}}^{(n,m)}) &= \sum_{\mathbf{a}} a_k\, p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,m)}), \\
&= \sum_{a \in \mathcal{A}} a\ p^{(n)}(a_k = a|\mathbf{r}, \hat{\mathbf{b}}^{(n,m)}), \qquad (7.84) \\
\rho(\mathbf{r}, \hat{\mathbf{b}}^{(n,m)}) &= \sum_{\mathbf{a}} \|a_k\|^2\, p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,m)}), \\
&= \sum_{a \in \mathcal{A}} \|a\|^2\ p^{(n)}(a_k = a|\mathbf{r}, \hat{\mathbf{b}}^{(n,m)}), \qquad (7.85)
\end{aligned}$$

where it is clear from (7.84)-(7.85) that $p^{(n)}(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n,m)})$ is computed as

$$p^{(n)}(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n,m)}) = \sum_{\mathbf{a}\setminus\{a_k\}} p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,m)}). \tag{7.86}$$

The vector of matched-filter outputs $\mathbf{y}$ is a sufficient statistic for symbol esti-

mation (see example 3.1) and therefore, we have

$$p^{(n)}(a_k|\mathbf{r},\hat{\mathbf{b}}^{(n,m)}) = \sum_{\mathbf{a}\backslash\{a_k\}} p^{(n)}(\mathbf{a}|\mathbf{y},\hat{\mathbf{b}}^{(n,m)}).$$
$$\propto \sum_{\mathbf{a}\backslash\{a_k\}} p(\mathbf{y}|\mathbf{a},\hat{\mathbf{b}}^{(n,m)})\, p_e^{(n)}(\mathbf{a}). \tag{7.87}$$

Taking then into account that the noise which affects the observations is white and that $p_e^{(n)}(\mathbf{a})$ factors as $p_e^{(n)}(\mathbf{a}) = \prod_k p_e^{(n)}(a_k)$, we have

$$p^{(n)}(a_k|\mathbf{r},\hat{\mathbf{b}}^{(n,m)}) \propto \sum_{\mathbf{a}\backslash\{a_k\}} \prod_{k'} p(y_{k'}|a_{k'},\hat{\mathbf{b}}^{(n,m)})\, p_e^{(n)}(a_{k'}),$$
$$\propto p(y_k|a_k,\hat{\mathbf{b}}^{(n,m)})\, p_e^{(n)}(a_k) \prod_{k'\neq k} \sum_{a_{k'}} p(y_{k'}|a_{k'},\hat{\mathbf{b}}^{(n,m)})\, p_e^{(n)}(a_{k'}). \tag{7.88}$$

The last factor in the right-hand side of (7.88) does not depend on a_k and we have therefore

$$p^{(n)}(a_k|\mathbf{r},\hat{\mathbf{b}}^{(n,m)}) \propto p(y_k|a_k,\hat{\mathbf{b}}^{(n,m)})\, p_e^{(n)}(a_k). \tag{7.89}$$

The evaluation of the (modified) a posteriori marginals $p^{(n)}(a_k|\mathbf{r},\hat{\mathbf{b}}^{(n,m)})$ is therefore straightforward since it only requires the multiplication of two known probabilities. Comparing with the implementation of the EM algorithm applied to the maximization of the actual likelihood function $p(\mathbf{r}|\mathbf{b})$ in section 7.4.1, we see that the implementation of the E-step is much easier in the context of the maximization of $p^{(n)}(\mathbf{r}|\mathbf{b})$. Indeed, as shown in section 7.4.3, the computation of marginals $p(a_k|\mathbf{r},\hat{\mathbf{b}})$ requires a cumbersome decoding operation. In contrast, $p^{(n)}(a_k|\mathbf{r},\hat{\mathbf{b}})$ may be computed by a simple multiplication. In this latter case, the complexity of the E-step no longer dominates the complexity of the EM algorithm.

The synchronization procedure deriving from the combination of the SP algorithm and the EM algorithm is illustrated in Algorithm 8. Notation $\hat{\mathbf{b}}^{(n,\infty)}$ denotes the estimate delivered after the converge of the EM algorithm at the n^{th} iteration of the SP algorithm. We note that since both the SP algorithm and the EM algorithm are iterative, we come up with a doubly-iterative synchronization procedure. In a first step, extrinsic probabilities $p_e^{(n)}(a_k)$ are computed using the previous estimate $\hat{\mathbf{b}}^{(n-1)}$. Then, a new estimate $\hat{\mathbf{b}}^{(n)}$ is computed by maximizing $p^{(n)}(\mathbf{r}|\mathbf{b})$ via an EM algorithm.

Algorithm 8: Synchronization based on the SP algorithm (implementation via the EM algorithm)

initialization:
  $\hat{\mathbf{b}}^{(0,0)} = \mathbf{b}_0$;
  `reset(`*factor-graph messages*`);`
repeat
  `compute(`$p_e^{(n)}(a_k)$`);`
  `compute(`$\hat{\mathbf{b}}^{(n)} = \arg\max_{\tilde{\mathbf{b}}} \log p^{(n)}(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}})$ `)` (EM algorithm);
    $\hat{\mathbf{b}}^{(n,0)} = \hat{\mathbf{b}}^{(n-1)}$;
    repeat
      `compute(`$p^{(n)}(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n,m)}) \propto p(y_k|a_k, \hat{\mathbf{b}}^{(n,m)})\, p_e^{(n-1)}(a_k)$`);`
      `compute(`$\hat{\mathbf{b}}^{(n,m+1)} = \arg\max_{\tilde{\mathbf{b}}} \mathcal{Q}^{(n)}(\mathbf{b} = \tilde{\mathbf{b}}, \hat{\mathbf{b}}^{(n,m)})$`);`
    until *convergence*;
    $\hat{\mathbf{b}}^{(n)} = \hat{\mathbf{b}}^{(n,\infty)}$;
until *convergence*;

The approximation of the EM synchronizer proposed by Noels et al. (see Algorithm 2) may actually be seen as an approximation of the SP-EM algorithm defined in Algorithm 8. Indeed, let us assume that only one EM iteration is performed at each SP iteration. We have therefore that $\hat{\mathbf{b}}^{(n)} = \hat{\mathbf{b}}^{(n,1)}$ and the function which is maximized at each SP iteration to get $\hat{\mathbf{b}}^{(n)}$ may be written as

$$\mathcal{Q}^{(n)}(\mathbf{b}, \hat{\mathbf{b}}^{(n-1)}) = \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) \log p(\mathbf{r}|\mathbf{a}, \mathbf{b}) \tag{7.90}$$

where

$$\begin{aligned} p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) &= p^{(n)}(\mathbf{a}|\mathbf{y}, \hat{\mathbf{b}}^{(n-1)}) \\ &\propto p(\mathbf{y}|\mathbf{a}, \hat{\mathbf{b}}^{(n-1)})\, p_e^{(n)}(\mathbf{a}). \end{aligned} \tag{7.91}$$

From (7.91), we see that marginals $p^{(n)}(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)})$ required to implement the EM synchronizer are then computed as

$$\begin{aligned} p^{(n)}(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) &= \sum_{\mathbf{a}\setminus\{a_k\}} p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)}), \\ &\propto p(y_k|a_k, \hat{\mathbf{b}}^{(n-1)})\, p_e^{(n)}(a_k). \end{aligned} \tag{7.92}$$

Now, (7.92) is equal to the product of the SP messages entering variable node a_k in the factor graph represented in Fig. 7.4, when $\hat{\mathbf{b}}^{(n)}$ is obtained at each iteration by maximizing (7.90). This exactly corresponds to the approximation

made by Noels et al. in Algorithm 2 to approximate the actual marginal probabilities $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$. The approach proposed by those authors may therefore be seen as an approximation of Algorithm 8 in which only one EM iteration is performed at each SP iteration.

We may ask the question about the validity of the approximation made by Noels et al. Actually, it seems quite clear that this approximation will be all the better as the EM algorithm converges rapidly to the maximum of $p^{(n)}(\mathbf{r}|\mathbf{b})$. Indeed, if the convergence is fast we may expect that $\hat{\mathbf{b}}^{(n,1)} \simeq \arg\max p^{(n)}(\mathbf{r}|\mathbf{b})$ whereas, on the contrary, $\hat{\mathbf{b}}^{(n,1)}$ may still be quite far from the maximum of $p^{(n)}(\mathbf{r}|\mathbf{b})$ when the convergence is slow. As mentioned in chapter 6, the speed of convergence of the EM algorithm is linear and may therefore be characterized by the rate of convergence. In section 9.2, we derive a lower bound on the mean convergence rate. In particular, we show that the convergence of the EM algorithm is a function of the amount of uncertainty about the transmitted symbols. In order to emphasize this result in a different way, let us rewrite the function which has to be maximized at each SP iteration as (see the appendix)

$$\log p^{(n)}(\mathbf{r}|\mathbf{b}) \propto \mathcal{Q}^{(n)}(\mathbf{b}, \hat{\mathbf{b}}^{(n,0)}) - \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,0)}) \log p^{(n)}(\mathbf{a}|\mathbf{r}, \mathbf{b}). \quad (7.93)$$

The first term in (7.93) is the function maximized at the first iteration of the EM algorithm. The second term may be regarded as a measure of the uncertainty about the transmitted symbols. In particular, note that if $\mathbf{b} = \hat{\mathbf{b}}^{(n,0)}$, this term is exactly equal to a conditional entropy [29], i.e.

$$H(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,0)}) = -\sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,0)}) \log p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,0)}).$$

Using (7.93), we see therefore that maximizing $\mathcal{Q}^{(n)}(\mathbf{b}, \hat{\mathbf{b}}^{(n,0)})$ is almost equivalent to maximizing $\log p^{(n)}(\mathbf{r}|\mathbf{b})$ as long as the second term in (7.93) remains small as compared to $\mathcal{Q}^{(n)}(\mathbf{b}, \hat{\mathbf{b}}^{(n,0)})$. This is for example the case when $p_e(\mathbf{a}) \simeq \delta(\mathbf{a} - \tilde{\mathbf{a}})$ since $-\sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,0)}) \log p^{(n)}(\mathbf{a}|\mathbf{r}, \mathbf{b})$ is then close to 0 and therefore

$$\log p^{(n)}(\mathbf{r}|\mathbf{b}) \simeq \mathcal{Q}^{(n)}(\mathbf{b}, \hat{\mathbf{b}}^{(n,0)}), \quad (7.94)$$

up to an irrelevant term. In order words, Noels et al's approximation holds when the extrinsic probabilities provide a lot of information about the transmitted symbols. This is for example the case in a turbo receiver after a sufficient number of turbo (or equivalently SP) iterations. On the contrary, during the first iterations the approximation proposed by Noels et al. might be rough and leads to poor results.

7.7.6 Combining the SP Algorithm and Gradient Algorithms

As mentioned above, the modified likelihood function $p^{(n)}(\mathbf{r}|\mathbf{b})$ has exactly the same structure as $p(\mathbf{r}|\mathbf{b})$ but uses $p_e^{(n)}(\mathbf{a})$ instead of $p(\mathbf{a})$ as sequence a priori information. As a consequence, the expressions of the gradient (7.41) and the Hessian matrix (7.49) remain valid by simply replacing $p(\mathbf{a})$ by $p_e^{(n)}(\mathbf{a})$, i.e.

$$\begin{aligned}
\nabla \log p^{(n)}(\mathbf{r}|\mathbf{b}) &= \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r},\mathbf{b}) \nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b}), \\
\nabla^2 \log p^{(n)}(\mathbf{r}|\mathbf{b}) &= \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r},\mathbf{b}) \nabla^2 \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \\
&\quad + \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r},\mathbf{b}) \left(\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \, \nabla^T \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \right) \\
&\quad - \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r},\mathbf{b}) \nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r},\mathbf{b}) \nabla^T \log p(\mathbf{r}|\mathbf{a},\mathbf{b}).
\end{aligned}$$

Since function $\log p(\mathbf{r}|\mathbf{a},\mathbf{b})$ does not change with respect to section 7.5, expressions (7.43)-(7.46) (resp. (7.51)) are still valid for the computation of the first (resp. second) derivatives of $\log p^{(n)}(\mathbf{r}|\mathbf{b})$. Probabilities $p^{(n)}(a_k|\mathbf{r},\mathbf{b})$ and $p^{(n)}(a_k, a_{k'}|\mathbf{r},\mathbf{b})$ have however to be computed by taking $p_e^{(n)}(\mathbf{a})$ instead of $p(\mathbf{a})$ into account. As shown in section 7.7.5, we have

$$p^{(n)}(a_k|\mathbf{r},\mathbf{b}) \propto p(y_k|a_k,\mathbf{b}) \, p_e^{(n)}(a_k). \tag{7.95}$$

Similarly, it is easy to show that $p^{(n)}(a_k, a_{k'}|\mathbf{r},\mathbf{b})$ may be computed as

$$p^{(n)}(a_k, a_{k'}|\mathbf{r},\mathbf{b}) \propto p(y_k|a_k,\mathbf{b}) \, p(y_{k'}|a_{k'},\mathbf{b}) \, p_e^{(n)}(a_k) \, p_e^{(n)}(a_{k'}). \tag{7.96}$$

The computation of probabilities $p^{(n)}(a_k|\mathbf{r},\mathbf{b})$ and $p^{(n)}(a_k, a_{k'}|\mathbf{r},\mathbf{b})$ is therefore straightforward since it only implies simple multiplications. Using (7.95) and (7.96), we also have

$$p^{(n)}(a_k, a_{k'}|\mathbf{r},\mathbf{b}) \propto p^{(n)}(a_k|\mathbf{r},\mathbf{b}) \, p^{(n)}(a_{k'}|\mathbf{r},\mathbf{b}), \tag{7.97}$$

i.e. joint probabilities $p^{(n)}(a_k, a_{k'}|\mathbf{r},\mathbf{b})$ may be evaluated from the knowledge of marginals $p^{(n)}(a_k|\mathbf{r},\mathbf{b})$ only. The synchronization algorithm based on the SP and the SD (resp. NR) algorithms is summarized in Algorithm 9 (resp. 10). Note that, since joint probabilities $p^{(n)}(a_k, a_{k'}|\mathbf{r},\mathbf{b})$ factor according to (7.97), (7.54) is no longer an approximation and the Hessian matrix can be properly evaluated thanks to (7.55).

Algorithm 9: Synchronization based on the SP algorithm (implementation via the SD algorithm)

initialization:
  $\hat{\mathbf{b}}^{(0,0)} = \mathbf{b}_0$;
  `reset(`*factor-graph messages*`);`
repeat
  `compute(`$p_e^{(n)}(a_k)$`);`
  `compute(`$\hat{\mathbf{b}}^{(n)} = \arg\max_{\tilde{\mathbf{b}}} \log p^{(n)}(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}})$ `)` (SD algorithm);
    $\hat{\mathbf{b}}^{(n,0)} = \hat{\mathbf{b}}^{(n-1)}$;
    repeat
      `compute(`$p^{(n)}(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n,m)}) \propto p(y_k|a_k, \hat{\mathbf{b}}^{(n,m)})\, p_e^{(n)}(a_k)$`);`
      `compute(`$\hat{\mathbf{b}}^{(n,m+1)} = \hat{\mathbf{b}}^{(n,m)} - \alpha^{(m)} \nabla \log p^{(n)}(\mathbf{r}|\hat{\mathbf{b}}^{(n,m)})$`);`
    until *convergence*;
    $\hat{\mathbf{b}}^{(n)} = \hat{\mathbf{b}}^{(n,\infty)}$;
until *convergence*;

Algorithm 10: Synchronization based on the SP algorithm (implementation via the NR algorithm)

initialization:
  $\hat{\mathbf{b}}^{(0,0)} = \mathbf{b}_0$;
  `reset(`*factor-graph messages*`);`
repeat
  `compute(`$p_e^{(n)}(a_k)$`);`
  `compute(`$\hat{\mathbf{b}}^{(n)} = \arg\max_{\tilde{\mathbf{b}}} \log p^{(n)}(\mathbf{r}|\mathbf{b} = \tilde{\mathbf{b}})$ `)` (NR algorithm);
    $\hat{\mathbf{b}}^{(n,0)} = \hat{\mathbf{b}}^{(n-1)}$;
    repeat
      `compute(`$p^{(n)}(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n,m)}) \propto p(y_k|a_k, \hat{\mathbf{b}}^{(n,m)})\, p_e^{(n)}(a_k)$`);`
      `compute(`$\hat{\mathbf{b}}^{(n,m+1)} = \hat{\mathbf{b}}^{(n,m)} - \left(\nabla^2 \log p^{(n)}(\mathbf{r}|\hat{\mathbf{b}}^{(n,m)})\right)^{-1} \nabla \log p^{(n)}(\mathbf{r}|\hat{\mathbf{b}}^{(n,m)})$`);`
    until *convergence*;
    $\hat{\mathbf{b}}^{(n)} = \hat{\mathbf{b}}^{(n,\infty)}$;
until *convergence*;

7.8 Main Points

In this chapter we focus on the design of code-aided (CA) ML synchronization algorithms dedicated to the estimation of continuous parameters. The considered algorithms operate in a *soft-decision-directed* (SDD) mode i.e. they use some symbol soft-information delivered by the data-detection part of the receiver to compute the synchronization estimate. We consider three different frameworks for SDD CA ML synchronization:

- The first SDD CA ML framework is based on the expectation-maximization (EM) algorithm (see section 7.4). We show that the implementation of an EM-based SDD synchronizer requires to compute at each iteration the symbol marginal probabilities $p(a_k|\mathbf{r},\mathbf{b})$ (see section 7.4.1). These probabilities are naturally available in state-of-the-art receivers based on a CMAP estimation of the transmitted data. The computation of marginals $p(a_k|\mathbf{r},\mathbf{b})$ dominates the complexity of the EM synchronizer and requires one data detection/decoding operation per EM iteration.

- A simplified version of the EM synchronizer is proposed by making an approximation on the a posteriori probabilities $p(a_k|\mathbf{r},\mathbf{b})$ (see section 7.4.3). Making this approximation, we show that the SSD phase synchronizer [15] proposed by Lottici and Luise in an ad-hoc fashion may be encompassed in the proposed EM synchronization framework.

- Another SDD CA ML framework is based on the gradient algorithm (see section 7.5). We consider two particular instances of the gradient algorithm: the steepest-descent (SD) algorithm (section 7.5.1) and the Newton-Raphson (NR) algorithm (section 7.5.3). The SD synchronizer requires to compute the gradient of $\log p(\mathbf{r}|\mathbf{b})$ whereas the NR synchronizer requires both the knowledge of the gradient and the Hessian matrix of $\log p(\mathbf{r}|\mathbf{b})$.

- The evaluation of the gradient of $\log p(\mathbf{r}|\mathbf{b})$ is shown to require the knowledge of the marginals $p(a_k|\mathbf{r},\mathbf{b})$. The computation of these probabilities dominates the complexity of the SD synchronizer: it implies one data detection/decoding operation per SD iteration. Similarly to the EM synchronizer, a simplified version of the SD synchronizer is derived by making an approximation on the a posteriori marginals $p(a_k|\mathbf{r},\mathbf{b})$. Note also that the performance of the SD synchronizer depends on the choice

of a stepsize factor (see section 7.5.2).

- The evaluation of the Hessian matrix of $\log p(\mathbf{r}|\mathbf{b})$ is shown to require both the knowledge of marginal probabilities $p(a_k|\mathbf{r},\mathbf{b})$ and joint probabilities $p(a_k, a_{k'}|\mathbf{r},\mathbf{b})$. The evaluation of the Hessian matrix is much more complex than the evaluation of the gradient. We propose however a tight approximation of the Hessian matrix which exhibits a computational order comparable to the evaluation of the gradient (see section 7.5.4).
- Finally, a synchronization framework based on the SP algorithm is derived (see section 7.7). In this framework, a new estimate is computed at each iteration by solving a modified ML problem: the modified likelihood function is built by considering the extrinsic probabilities delivered by the symbol-detection part of the receiver as a priori probabilities. Unlike the EM, SD or NR synchronizers, we cannot prove that the convergence of the SP synchronizer is ensured. However, as long as the SP synchronizer converges, we show that its convergence point is a stationary point of the actual likelihood function.
- The ML synchronization frameworks based on the EM and the gradient algorithms can be applied to the resolution of the modified ML problem arising in the SP synchronization framework. Doing so, it turns out that the approximated version of the EM (resp. SD and NR) synchronizer mentioned above is only a particular case of a more general synchronization algorithm based on both the SP and the EM (resp. SD and NR) algorithms.

7.9 Appendices

7.9.1 Proof of (7.41)

In this appendix, we show that

$$\nabla \log p(\mathbf{r}|\mathbf{b}) = E_{\mathbf{a}|\mathbf{r},\mathbf{b}}\Big[\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\Big]. \tag{7.98}$$

Let us first notice that $\nabla \log p(\mathbf{r}|\mathbf{b})$ may also be expressed as

$$\nabla \log p(\mathbf{r}|\mathbf{b}) = \frac{1}{p(\mathbf{r}|\mathbf{b})} \nabla p(\mathbf{r}|\mathbf{b}) \tag{7.99}$$

Considering that $p(\mathbf{r}|\mathbf{b}) = \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{a})$, we have

$$\nabla \log p(\mathbf{r}|\mathbf{b}) = \sum_{\mathbf{a}} \frac{p(\mathbf{a})}{p(\mathbf{r}|\mathbf{b})} \nabla p(\mathbf{r}|\mathbf{a},\mathbf{b}). \tag{7.100}$$

Taking into account that $\nabla p(\mathbf{r}|\mathbf{a},\mathbf{b}) = p(\mathbf{r}|\mathbf{a},\mathbf{b})\,\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b})$, (7.100) becomes

$$\nabla \log p(\mathbf{r}|\mathbf{b}) = \sum_{\mathbf{a}} \frac{p(\mathbf{r}|\mathbf{a},\mathbf{b})\,p(\mathbf{a})}{p(\mathbf{r}|\mathbf{b})}\,\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b}). \tag{7.101}$$

Now, $\frac{p(\mathbf{r}|\mathbf{a},\mathbf{b})p(\mathbf{a})}{p(\mathbf{r}|\mathbf{b})} = p(\mathbf{a}|\mathbf{r},\mathbf{b})$ and we get therefore (7.98).

7.9.2 Proof of (7.49)

In this appendix we show that the Hessian matrix of $\log p(\mathbf{r}|\mathbf{b})$ may be expressed as

$$\begin{aligned}\nabla^2 \log p(\mathbf{r}|\mathbf{b}) = {} & E_{\mathbf{a}|\mathbf{r},\mathbf{b}}\Big[\nabla^2 \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\Big] \\ & + E_{\mathbf{a}|\mathbf{r},\mathbf{b}}\Big[\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\,\nabla^T \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\Big] \\ & - E_{\mathbf{a}|\mathbf{r},\mathbf{b}}\Big[\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\Big]\; E_{\mathbf{a}|\mathbf{r},\mathbf{b}}\Big[\nabla^T \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\Big].\end{aligned} \tag{7.102}$$

Let us consider the element at the i^{th} row and j^{th} column of $\nabla^2 \log p(\mathbf{r}|\mathbf{b})$, i.e.

$$\Big(\nabla^2 \log p(\mathbf{r}|\mathbf{b})\Big)_{i,j} = \frac{\partial}{\partial b_j}\left(E_{\mathbf{a}|\mathbf{r},\mathbf{b}}\Big[\frac{\partial}{\partial b_i}\log p(\mathbf{r}|\mathbf{a},\mathbf{b})\Big]\right), \tag{7.103}$$

where equality (7.103) follows from (7.41). Applying the chain rule of the derivation operator, we have

$$\begin{aligned}\Big(\nabla^2 \log p(\mathbf{r}|\mathbf{b})\Big)_{i,j} = {} & \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r},\mathbf{b})\,\frac{\partial^2 \log p(\mathbf{r}|\mathbf{a},\mathbf{b})}{\partial b_i\,\partial b_j} \\ & + \sum_{\mathbf{a}} \frac{\partial p(\mathbf{a}|\mathbf{r},\mathbf{b})}{\partial b_j}\,\frac{\partial \log p(\mathbf{r}|\mathbf{a},\mathbf{b})}{\partial b_i}\end{aligned} \tag{7.104}$$

Factor $\frac{\partial p(\mathbf{a}|\mathbf{r},\mathbf{b})}{\partial b_j}$ may be further expanded as

$$\frac{\partial p(\mathbf{a}|\mathbf{r},\mathbf{b})}{\partial b_j} = p(\mathbf{a}|\mathbf{r},\mathbf{b})\,\frac{\partial \log p(\mathbf{a}|\mathbf{r},\mathbf{b})}{\partial b_j}.$$

Taking into account that $p(\mathbf{a}|\mathbf{r},\mathbf{b}) = \frac{p(\mathbf{r}|\mathbf{a},\mathbf{b})p(\mathbf{a})}{p(\mathbf{r}|\mathbf{b})}$ and neglecting the terms independent of $\mathbf{b}$, we have

$$\frac{\partial p(\mathbf{a}|\mathbf{r},\mathbf{b})}{\partial b_j} = p(\mathbf{a}|\mathbf{r},\mathbf{b})\left(\frac{\partial \log p(\mathbf{r}|\mathbf{a},\mathbf{b})}{\partial b_j} - \frac{\partial \log p(\mathbf{r}|\mathbf{b})}{\partial b_j}\right). \tag{7.105}$$

Plugging (7.105) into (7.104) and using the fact that

$$\frac{\partial}{\partial b_j}\log p(\mathbf{r}|\mathbf{b}) = \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r},\mathbf{b})\,\frac{\partial}{\partial b_j}\log p(\mathbf{r}|\mathbf{a},\mathbf{b})$$

from (7.41), we have

$$\begin{aligned}\left(\nabla^2 \log p(\mathbf{r}|\mathbf{b})\right)_{i,j} &= \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r},\mathbf{b}) \frac{\partial^2 \log p(\mathbf{r}|\mathbf{a},\mathbf{b})}{\partial b_i \, \partial b_j} \\ &+ \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r},\mathbf{b}) \left(\frac{\partial \log p(\mathbf{r}|\mathbf{a},\mathbf{b})}{\partial b_i} \frac{\partial \log p(\mathbf{r}|\mathbf{a},\mathbf{b})}{\partial b_j} \right) \\ &- \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r},\mathbf{b}) \frac{\partial \log p(\mathbf{r}|\mathbf{a},\mathbf{b})}{\partial b_i} \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r},\mathbf{b}) \frac{\partial \log p(\mathbf{r}|\mathbf{a},\mathbf{b})}{\partial b_j},\end{aligned}$$

which proves (7.102).

7.9.3 Proof of (7.77)

In this appendix we show that

$$\nabla \log p^{(n)}(\mathbf{r}|\hat{\mathbf{b}}^{(n-1)}) = \nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n-1)}), \tag{7.106}$$

as long as the factor graph used to compute extrinsic probabilities $p_e(a_k)$ is cycle free.

Using expression (7.41), we have

$$\nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n-1)}) = \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r},\hat{\mathbf{b}}^{(n-1)}) \nabla \log p(\mathbf{r}|\mathbf{a},\hat{\mathbf{b}}^{(n-1)}). \tag{7.107}$$

In section 7.4.1, we show that $\log p(\mathbf{r}|\mathbf{a},\hat{\mathbf{b}}^{(n-1)})$ can be expressed as a product of functions which only depend on one symbol a_k, i.e.

$$\log p(\mathbf{r}|\mathbf{a},\hat{\mathbf{b}}^{(n-1)}) = \prod_k f_k(a_k,\hat{\mathbf{b}}^{(n-1)}). \tag{7.108}$$

Plugging (7.108) in (7.107), we have

$$\begin{aligned}\nabla \log p(\mathbf{r}|\hat{\mathbf{b}}^{(n-1)}) &= \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r},\hat{\mathbf{b}}^{(n-1)}) \nabla \log\Big(\prod_k f_k(a_k,\hat{\mathbf{b}}^{(n-1)})\Big), \\ &= \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{r},\hat{\mathbf{b}}^{(n-1)}) \sum_k \nabla \log f_k(a_k,\hat{\mathbf{b}}^{(n-1)}), \\ &= \sum_k \sum_{a_k} p(a_k|\mathbf{r},\hat{\mathbf{b}}^{(n-1)}) \nabla \log f_k(a_k,\hat{\mathbf{b}}^{(n-1)}).\end{aligned} \tag{7.109}$$

Using expression (7.41), we may also write that

$$\nabla \log p^{(n)}(\mathbf{r}|\hat{\mathbf{b}}^{(n-1)}) = \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r},\hat{\mathbf{b}}^{(n-1)}) \nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b}^{(n-1)}). \tag{7.110}$$

Plugging (7.108) in (7.110), we obtain

$$\nabla \log p^{(n)}(\mathbf{r}|\hat{\mathbf{b}}^{(n-1)}) = \sum_k \sum_{a_k} p^{(n)}(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) \nabla \log f_k(a_k, \hat{\mathbf{b}}^{(n-1)}). \quad (7.111)$$

where

$$p^{(n)}(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) \propto p(y_k|a_k, \hat{\mathbf{b}}^{(n-1)})\, p_e^{(n)}(a_k). \quad (7.112)$$

Now, if the factor graph used to compute extrinsic probability $p_e^{(n)}(a_k)$ is cycle free, the actual marginals $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)})$ are also equal to (7.112) (see chapter 5). Therefore, (7.111) is identical to (7.109), which proves (7.107).

7.9.4 Proof of (7.93)

In this appendix, we show that $p^{(n)}(\mathbf{r}|\mathbf{b})$ may be expanded as

$$\log p^{(n)}(\mathbf{r}|\mathbf{b}) \propto \mathcal{Q}^{(n)}(\mathbf{b}, \hat{\mathbf{b}}^{(n,0)}) - \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,0)}) \log p^{(n)}(\mathbf{a}|\mathbf{r}, \mathbf{b}), \quad (7.113)$$

where

$$\mathcal{Q}^{(n)}(\mathbf{b}, \hat{\mathbf{b}}^{(n,0)}) = \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,0)}) \log p(\mathbf{r}|\mathbf{a}, \mathbf{b}). \quad (7.114)$$

Indeed, using Bayes rule we have that

$$p^{(n)}(\mathbf{r}|\mathbf{b}) = \frac{p(\mathbf{r}|\mathbf{a}, \mathbf{b})\, p_e^{(n)}(\mathbf{a})}{p^{(n)}(\mathbf{a}|\mathbf{r}, \mathbf{b})}. \quad (7.115)$$

Taking the logarithm of (7.115) and dropping the term independent of $\mathbf{b}$, we obtain

$$\log p^{(n)}(\mathbf{r}|\mathbf{b}) \propto \log p(\mathbf{r}|\mathbf{a}, \mathbf{b}) - \log p^{(n)}(\mathbf{a}|\mathbf{r}, \mathbf{b}). \quad (7.116)$$

Taking the expectation of the two sides of (7.116) with respect to $p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,0)})$, we have

$$\begin{aligned} \log p^{(n)}(\mathbf{r}|\mathbf{b}) \propto & \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,0)}) \log p(\mathbf{r}|\mathbf{a}, \mathbf{b}) \\ & - \sum_{\mathbf{a}} p^{(n)}(\mathbf{a}|\mathbf{r}, \hat{\mathbf{b}}^{(n,0)}) \log p^{(n)}(\mathbf{a}|\mathbf{r}, \mathbf{b}), \end{aligned} \quad (7.117)$$

which proves (7.113).

Code-Aided Synchronization of Discrete Parameters

8

In this chapter, we address the problem of code-aided (CA) synchronization of discrete-valued parameters. This problem is also often referred to as CA *hypothesis testing*. The most common examples of hypothesis-testing issues in burst communication systems include phase and timing ambiguity resolution, i.e. the estimation of k_θ and k_τ (see chapter 2). In this chapter, we will exclusively focus on these problems. Other synchronization parameters will be assumed perfectly-known in the remainder of the chapter. We also refer the interested reader to the excellent work by Wymeersch [19].

The chapter is organized as follows. In the first section, we review some methods which have already been proposed in the literature to solve hypothesis-testing problems. Two main approaches may be distinguished among these "conventional" methods: a first family of methods operates in a DA mode whereas the second one operates in a CA mode. On the one hand, the DA family is shown to be inadequate in state-of-the-art receivers operating at low SNRs. On the other hand, the CA methods proposed so far have all been derived (to some extent) in an ad-hoc fashion. The remainder of the chapter is therefore dedicated to the derivation of more rigorous solutions for efficiently estimating discrete-valued parameters. We first place CA synchronization in the context of MAP estimation. A simplified version of the MAP synchronizer is proposed and motivated. Then, an algorithm based on the factor-graph representation and the SP algorithm is proposed to efficiently compute the exact MAP solution. In a second part of this chapter, we derive discrete-parameter estimators based on "data-detection-oriented" criteria: the estimators are designed to minimize the probability of committing an error

on the subsequent decision made on the transmitted data. Based on this approach, estimators designed to minimize the FER and BER are proposed.

8.1 Conventional Approaches

Conventional approaches to deal with discrete-valued parameter estimation are based on the transmission of pilot symbols and rest on the MAP or ML criterion[1]. The two most popular data-aided (DA) hypothesis testing methods are Massey [70] and Lui [71]. In these references, the authors derive the maximum-likelihood (ML) DA hypothesis-testing procedure when the data symbols are assumed to be uncoded.

Since it implies the transmission of some pilot sequences, DA hypothesis testing leads to both a waste of power and bandwidth. In communication systems using powerful error-correcting codes, the low signal-to-noise ratio (SNR) encountered by conventional DA hypothesis-testing devices may require the transmission of an unacceptable number of pilot symbols. This problem has led several authors to propose hypothesis-testing procedures which take into account the structure of the code used to transmit the data, see e.g. [72], [73], [74], [17], [75], [18]. Among these methods, we can distinguish between two main approaches: the authors either propose hypothesis-testing methods based on the fact that some decoder metrics vary as a function of the considered hypotheses ([72], [73], [74], [17]) or place the coded-aided (CA) hypothesis-testing problem in the context of MAP (or ML) estimation ([75], [18]).

In [73] for example, the authors base their synchronization method on the syndrome of a convolutional code: if the system is synchronized, the parity-check syndrome of the code is generated; otherwise a random sequence is generated. The decision about the synchronization-parameter value is therefore made according to whether the true syndrome or a random sequence is observed. In [17], the authors estimate the synchronization parameter by observing, under different hypotheses, the distribution of the a posteriori log-likelihood ratios (LLRs) delivered by the decoder: the distance between the two modes of the distribution should be maximized when the system is synchronized. The authors referred to their method as "Mode

[1] depending on whether a priori information on $\mathbf{b}$ is available or not.

Separation".

In [75] and [18], [76], the authors consider the CA hypothesis-testing problem in the context of MAP estimation. Since, the exact MAP solution is assumed intractable[2], different suboptimal MAP-based solutions are proposed. In [75], the authors modify the likelihood function by only keeping its largest term and decide whether the current estimate is true or not by a threshold decision on this modified likelihood function. In [18], another approach is followed: the authors place the MAP estimation problem in the context of the EM algorithm (see chapter 6). Since the EM algorithm is not suited to discrete estimation problems, the authors propose some arbitrary modifications which are shown to give good results in practice.

Note that most of the CA synchronization algorithms described above have the common feature[3] to require one decoding operation per hypothesis to test. This characteristic is actually a direct consequence of the discrete nature of the problem in hand. We will see that this feature is also present in the synchronizers which will be derived in the remainder of this chapter. The complexity associated with CA hypothesis testing algorithms can however be decreased by considering *list synchronization* [77]-[78]. This method actually initiates by generating a list of the v most probable candidates using a simple algorithm such as [70], [71]. It then constrains the complex CA synchronizer to only test the q most probable decisions which are in the list instead of all the $|\mathcal{B}|$ possibilities. The overall synchronization complexity is therefore reduced by a factor $\frac{q}{|\mathcal{B}|}$.

8.2 MAP Synchronization

When the parameters to estimate are discrete-valued, MAP estimation is the optimal solution in terms of the minimization of the probability of wrong decision (see chapter 3). Considering the following MAP estimation problem,

$$\hat{\mathbf{b}}_{MAP} = \arg\max_{\tilde{\mathbf{b}} \in \mathcal{B}} p(\mathbf{b} = \tilde{\mathbf{b}}|\mathbf{r}), \tag{8.1}$$

we are therefore ensured to make the decision about $\mathbf{b}$ which minimizes the probability of committing an error. Unfortunately, the objective function

[2] We will show in the sequel that this is not always true.

[3] As long as $|\mathcal{B}| > 2$, where $\mathcal{B}$ is the set of possible values of the synchronization parameter.

$p(\mathbf{b}|\mathbf{r})$ contains a huge number of terms, i.e.

$$p(\mathbf{b}|\mathbf{r}) = \sum_{\mathbf{a}\in\mathcal{A}^K} p(\mathbf{b}|\mathbf{r},\mathbf{a})\, p(\mathbf{a}|\mathbf{r}), \tag{8.2}$$

and appears therefore quite difficult to maximize. As a consequence, "MAP" synchronization methods proposed in the literature, see e.g. [75], [18], are rather approximations of the actual MAP solution (8.1).

In this section, we derive MAP(-based) algorithms for the estimation of discrete synchronization parameters. In a first subsection, we propose to modify goal function $p(\mathbf{b}|\mathbf{r})$ by taking into account the probability of making a wrong decision about the data sequence. The idea of this approach is as follows: when we have little hope of reliably decoding $\mathbf{u}$ even when $\mathbf{b}$ is known, we do not care to correctly estimate $\mathbf{b}$. As we will see, this observation enables to simplify the implementation of the MAP criterion. In the second subsection, we place the MAP estimation problem in the factor-graph framework: we show that the objective function $p(\mathbf{b}|\mathbf{r})$ may be computed with a tractable complexity by means of the SP algorithm. As a consequence, we stress that the actual MAP solution (8.1) may be computed with a computational complexity comparable (even lower sometimes!) to other conventional CA methods.

8.2.1 Data-oriented MAP-based Synchronization

In this subsection we propose a simplified version of the MAP synchronizer[4]. The main idea of the proposed approach is to allow the estimator to make a non-MAP decision about the synchronization parameter when there is anyway little hope to correctly recover the transmitted data with a proper synchronization.

In order to simplify the MAP estimation problem, let us rewrite objective function $p(\mathbf{b}|\mathbf{r})$ in a different way. Using Bayes rule, this function may be reformulated as

$$p(\mathbf{b}|\mathbf{r}) = \frac{p(\mathbf{a})\, p(\mathbf{b})\, p(\mathbf{r}|\mathbf{a},\mathbf{b})}{p(\mathbf{a}|\mathbf{r},\mathbf{b})\, p(\mathbf{r})}. \tag{8.3}$$

Taking the logarithm and dropping irrelevant terms, we obtain

$$\log p(\mathbf{b}|\mathbf{r}) \propto \log p(\mathbf{b}) + \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) - \log p(\mathbf{a}|\mathbf{r},\mathbf{b}). \tag{8.4}$$

[4]This work results from a joint collaboration with H. Wymeersch and F. Simoens from the Ghent university. Thanks a lot to them!

Taking the expectation of both sides of (8.4) with respect to $p(\mathbf{a}|\mathbf{r},\mathbf{b})$ leads to[5]

$$\log p(\mathbf{b}|\mathbf{r}) \propto \log p(\mathbf{b}) + \mathcal{Q}(\mathbf{b}) + H(\mathbf{b}), \tag{8.5}$$

where

$$\mathcal{Q}(\mathbf{b}) \triangleq \sum_{\tilde{\mathbf{a}}\in\mathcal{A}^K} p(\mathbf{a}=\tilde{\mathbf{a}}|\mathbf{r},\mathbf{b}) \log p(\mathbf{r}|\mathbf{a}=\tilde{\mathbf{a}},\mathbf{b}), \tag{8.6}$$

$$H(\mathbf{b}) \triangleq \sum_{\tilde{\mathbf{a}}\in\mathcal{A}^K} p(\mathbf{a}=\tilde{\mathbf{a}}|\mathbf{r},\mathbf{b}) \log p(\mathbf{a}=\tilde{\mathbf{a}}|\mathbf{r},\mathbf{b}). \tag{8.7}$$

We can make the following observation regarding the three terms in (8.5):

- $\log p(\mathbf{b})$ is known at the receiver.
- $\mathcal{Q}(\mathbf{b})$ has exactly the same structure as the function maximized at each iteration by the EM synchronizer derived in chapter 7. Due to the particular structure of the received signal (2.8), it turns out that $\mathcal{Q}(\mathbf{b})$ can be computed based on the knowledge of marginal a posteriori probabilities $p(a_k|\mathbf{r},\mathbf{b})$. The computation of those marginals has been discussed in chapter 7: they are usually available (at least in approximation) in most state-of-the-art receivers.
- $H(\mathbf{b})$ can be interpreted as an entropy of the data sequence, given an observation $\mathbf{r}$ and parameter $\mathbf{b}$. We have therefore

  $$0 \leq H(\mathbf{b}) \leq \log|\mathcal{A}^K|. \tag{8.8}$$

  Unlike $\mathcal{Q}(\mathbf{b})$, $H(\mathbf{b})$ does not seem to benefit from obvious computational simplifications. Therefore, the evaluation of this term makes the computation of $p(\mathbf{b}|\mathbf{r})$ quite difficult.

On the one hand, we see that $H(\mathbf{b})$ seems to make the implementation of the MAP synchronizer difficult. On the other hand, we may ask the question whether we always need an estimator as powerful as the MAP estimator. Indeed, we only need a correct estimate of the synchronization parameters when we can subsequently reliably recover the transmitted data sequence. In all other cases, since we have little hope of recovering the data, we do not really care what the estimate of the synchronization parameters is.

Based on this idea, let us consider the probability of committing an error on the transmitted sequence when we consider a MAP synchronizer and a

[5] We have assumed that $p(\mathbf{a})$ is uniform.

CMAP sequence estimator (4.12). Denoting by $p(e_{\mathbf{u}}|\mathbf{b} = \mathbf{b}^\star)$ the probability of wrong sequence detection when synchronization parameter is equal to $\mathbf{b}^\star$, we show in the appendix that

$$p(e_{\mathbf{u}}|\mathbf{b} = \mathbf{b}^\star) = 1 - \int_{\mathcal{R}_{\mathbf{b}}} \max_{\tilde{\mathbf{a}} \in \mathcal{A}^K} \{p(\mathbf{a} = \tilde{\mathbf{a}}|\mathbf{r}, \mathbf{b} = \mathbf{b}^\star)\}\, p(\mathbf{r}|\mathbf{b} = \mathbf{b}^\star)\, d\mathbf{r}, \tag{8.9}$$

where $\mathcal{R}_{\mathbf{b}}$ denotes the domain on which the MAP synchronizer is able to correctly recover $\mathbf{b}^\star$. From (8.9) we see that a "good" suboptimal synchronizer should be able to correctly estimate $\mathbf{b}^\star$ $\forall\, \mathbf{r} \in \mathcal{R}_{\mathbf{b}}$ such that $\max_{\tilde{\mathbf{a}} \in \mathcal{A}^K} \{p(\mathbf{a} = \tilde{\mathbf{a}}|\mathbf{b} = \mathbf{b}^\star, \mathbf{r})\}$ is large. This occurs when $p(\mathbf{a} = \tilde{\mathbf{a}}|\mathbf{b} = \mathbf{b}^\star, \mathbf{r})$ has a single pronounced peak. On the other hand, when $\max_{\tilde{\mathbf{a}} \in \mathcal{A}^K} \{p(\mathbf{a} = \tilde{\mathbf{a}}|\mathbf{b} = \mathbf{b}^\star, \mathbf{r})\}$ is small, this will not significantly impact $p(e_{\mathbf{u}}|\mathbf{b})$, so it is irrelevant to properly synchronize the system.

As a consequence, we are interested in making the same decision as the actual MAP estimator $\forall\, \mathbf{r} \in \mathcal{R}_{\mathbf{b}}$ when $p(\mathbf{a}|\mathbf{r}, \mathbf{b})$ has a single pronounced peak. In other words, when there exists a unique sequence $\check{\mathbf{a}}$ such that $\forall\, \tilde{\mathbf{a}} \neq \check{\mathbf{a}}$:

$$p(\mathbf{a} = \check{\mathbf{a}}|\mathbf{r}, \mathbf{b} = \mathbf{b}^\star) \gg p(\mathbf{a} = \tilde{\mathbf{a}}|\mathbf{r}, \mathbf{b} = \mathbf{b}^\star). \tag{8.10}$$

Taking (8.10) into account, the expression of the MAP synchronizer may be simplified. Indeed, we have by definition of the MAP criterion that

$$\log p(\mathbf{b} = \hat{\mathbf{b}}_{MAP}|\mathbf{r}) \geq \log p(\mathbf{b} = \tilde{\mathbf{b}}|\mathbf{r}) \qquad \forall\, \tilde{\mathbf{b}} \neq \hat{\mathbf{b}}_{MAP}. \tag{8.11}$$

Using (8.5), this inequality may also be reformulated as

$$\Delta(\hat{\mathbf{b}}_{MAP}) - \Delta(\tilde{\mathbf{b}}) \geq H(\tilde{\mathbf{b}}) - H(\hat{\mathbf{b}}_{MAP}), \tag{8.12}$$

where

$$\Delta(\mathbf{b}) \triangleq \log p(\mathbf{b}) + \mathcal{Q}(\mathbf{b}). \tag{8.13}$$

By the definition of $\mathcal{R}_{\mathbf{b}}$, we have

$$H(\hat{\mathbf{b}}_{MAP}) = H(\mathbf{b}^\star). \tag{8.14}$$

Using (8.10), we have furthermore

$$H(\mathbf{b}^\star) \simeq 0. \tag{8.15}$$

Indeed, (8.10) implies that the uncertainty on $\mathbf{a}$ given $\mathbf{r}$ and $\mathbf{b} = \mathbf{b}^\star$ is small and therefore, so is the entropy. Considering then (8.8), we finally get

$$\Delta(\hat{\mathbf{b}}_{MAP}) \geq \Delta(\tilde{\mathbf{b}}) \qquad \forall\, \tilde{\mathbf{b}} \neq \hat{\mathbf{b}}_{MAP}, \tag{8.16}$$

or in other words,

$$\hat{\mathbf{b}}_{MAP} = \arg\max_{\tilde{\mathbf{b}}\in\mathcal{B}} \Delta(\tilde{\mathbf{b}}). \tag{8.17}$$

The interpretation is as follows:

- Suppose (8.10) is satisfied. This means we can reliably decode the data, when we can recover the correct value of $\mathbf{b}$. For any $\mathbf{b}$, the MAP estimator (8.1) has a corresponding $\mathcal{R}_{\mathbf{b}}$. When $\mathbf{r} \in \mathcal{R}_{\mathbf{b}}$, the MAP estimator can correctly recover $\mathbf{b}$. Combining these statements, we have just shown that in this case the estimator (8.17) will also produce the correct value of $\mathbf{b}$. When $\mathbf{r} \notin \mathcal{R}_{\mathbf{b}}$, the MAP estimator (8.1) will give an incorrect estimate of $\mathbf{b}$, leading with high probability to a decoding error. In such cases, using (8.17) instead of (8.1) will not lead to any significant degradation in terms of the error probability.
- On the other hand, when (8.10) is not satisfied, we have little hope of reliably decoding $\mathbf{a}$, even when $\mathbf{b}$ is known. Our design rule then tells us that we do not care whether or not we estimate $\mathbf{b}$ correctly. This means that we can use (8.1) just as well as (8.17), without any significant impact on the error probability $p(e_{\mathbf{u}}|\mathbf{b})$.

In conclusion, using (8.17) as opposed to (8.1) as a practical technique to determine $\mathbf{b}$ should not greatly affect the frame error rate performance. Interestingly, this criterion was obtained in an ad-hoc fashion in [18].

8.2.2 MAP synchronization via the SP algorithm

In this subsection we show that the objective function which has to be maximized in the MAP estimation problem (8.1) may be efficiently computed by means of the factor-graph representation and the SP algorithm.

Let us consider the factorization of $p(\mathbf{b}|\mathbf{r})$. Using Bayes rule and dropping the terms independent of $\mathbf{b}$, we have

$$\begin{aligned} p(\mathbf{b}|\mathbf{r}) &\propto p(\mathbf{b})\, p(\mathbf{r}|\mathbf{b}), \\ &\propto p(\mathbf{b}) \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{a}). \end{aligned}$$

The marginalization with respect to $\mathbf{a}$ appears to be the critical point in the computation of $p(\mathbf{b}|\mathbf{r})$: the summation may contain a huge number of terms

if we do not properly exploit the particular factorization of $p(\mathbf{r}|\mathbf{a},\mathbf{b})\,p(\mathbf{a})$. Let us therefore focus more particularly on the factorization of $p(\mathbf{r}|\mathbf{a},\mathbf{b})$ and $p(\mathbf{a})$. First, note that

$$p(\mathbf{r}|\mathbf{a},\mathbf{b}) \propto C(\mathbf{b})\, p(\mathbf{y}(\mathbf{b})|\mathbf{a},\mathbf{b}), \tag{8.18}$$

where

$$C(\mathbf{b}) \triangleq \exp\Big(\frac{1}{2N_0}\sum_{k=0}^{K-1} \|y_k(\tau = k_\tau,\nu)\|^2\Big) \tag{8.19}$$

and $\mathbf{y} = [y_0(\tau,\nu), y_1(\tau,\nu), \ldots y_{K-1}(\tau,\nu)]^T$ denotes the vector of matched-filter outputs (see (3.26)). Indeed, we have from (3.28)-(3.30) that $p(\mathbf{r}|\mathbf{a},\mathbf{b})$ may be written as

$$p(\mathbf{r}|\mathbf{a},\mathbf{b}) \propto \exp\Big(-\frac{T_s\,\sum_l \|r(lT_s)\|^2}{2N_0}\Big)\, g(\mathbf{y},\mathbf{b}), \tag{8.20}$$

where

$$g(\mathbf{y},\mathbf{b}) = \exp\left(-\frac{\sum_k \|\, A\, a_k\,\|^2}{2N_0}\right) \times \exp\left(\frac{2\sum_k \mathrm{Re}\big\{A\, a_k^*\, e^{-j\theta}\, y_k(\nu,\tau)\big\}}{2N_0}\right) \tag{8.21}$$

On the other hand, $p(\mathbf{y}|\mathbf{a},\mathbf{b})$ is equal to

$$p(\mathbf{y}(\mathbf{b})|\mathbf{a},\mathbf{b}) \propto \exp\Big(-\frac{\sum_k \|y_k(\tau,\nu)\|^2}{2N_0}\Big)\, g(\mathbf{y},\mathbf{b}). \tag{8.22}$$

Comparing (8.20) with (8.22), we have

$$p(\mathbf{r}|\mathbf{a},\mathbf{b}) \propto \exp\Big(-\frac{T_s\,\sum_l \|r(lT_s)\|^2}{2N_0}\Big)\, \exp\Big(\frac{\sum_k \|y_k(\tau,\nu)\|^2}{2N_0}\Big)\, p(\mathbf{y}(\mathbf{b})|\mathbf{a},\mathbf{b}). \tag{8.23}$$

Since the first factor in (8.23) depends neither on $\mathbf{a}$ nor $\mathbf{b}$, it can be dropped and we come up with (8.18).

Using (8.18) and taking into account that $C(\mathbf{b})$ does not depend on $\mathbf{a}$, we can therefore write

$$p(\mathbf{b}|\mathbf{r}) \propto p(\mathbf{b})\, C(\mathbf{b}) \sum_{\mathbf{a}} p(\mathbf{y}|\mathbf{a},\mathbf{b})\, p(\mathbf{a}).$$

Taking into account that the noise which affects the observations is white, we have on the one hand

$$p(\mathbf{y}|\mathbf{a},\mathbf{b}) = \prod_k p(y_k|\mathbf{a},\mathbf{b}). \tag{8.24}$$

On the other hand, factor $p(\mathbf{a})$ may be factorized as

$$\begin{aligned} p(\mathbf{a}) &= \sum_{\sim\{\mathbf{a}\}} p(\mathbf{a},\mathbf{x},\mathbf{u}), \\ &= \sum_{\sim\{\mathbf{a}\}} p(\mathbf{a}|\mathbf{x})\, p(\mathbf{x}|\mathbf{u})\, p(\mathbf{u}). \end{aligned} \tag{8.25}$$

The factorization of $p(\mathbf{a}|\mathbf{x})$ and $p(\mathbf{x}|\mathbf{u})$ respectively depends on the bit-to-symbol mapping and the error-correcting code used for the transmission (see chapter 5). This all leads to

$$p(\mathbf{b}|\mathbf{r}) \propto \sum_{\sim\{\mathbf{b}\}} p(\mathbf{b})\, C(\mathbf{b}) \prod_k p(y_k|a_k,\mathbf{b})\, p(\mathbf{a}|\mathbf{x})\, p(\mathbf{x}|\mathbf{u})\, p(\mathbf{u}). \tag{8.26}$$

The factor graph associated to (8.26) is represented in Fig. 8.1. Due to the dependence of coded bits, the factor graph necessarily contains cycles and we are therefore no longer ensured of the accuracy of the results delivered by the SP algorithm. This can actually be avoided by rewriting (8.26) in a slightly different way:

$$p(\mathbf{b}|\mathbf{r}) \propto p(\mathbf{b})\, C(\mathbf{b}) \sum_{u_l} \sum_{\sim\{u_l\}\setminus\{\mathbf{b}\}} \prod_k p(y_k|a_k,\mathbf{b})\, p(\mathbf{a}|\mathbf{x})\, p(\mathbf{x}|\mathbf{u})\, p(\mathbf{u}). \tag{8.27}$$

Now,

$$\sum_{\sim\{u_l\}\setminus\{\mathbf{b}\}} \prod_k p(y_k|a_k,\mathbf{b})\, p(\mathbf{a}|\mathbf{x})\, p(\mathbf{x}|\mathbf{u})\, p(\mathbf{u}) \propto p(u_l,\mathbf{y}|\mathbf{b}), \tag{8.28}$$

and therefore

$$p(\mathbf{b}|\mathbf{r}) \propto p(\mathbf{b})\, C(\mathbf{b}) \sum_{u_l} p(u_l,\mathbf{y}|\mathbf{b}). \tag{8.29}$$

Note that the evaluation of $p(\mathbf{b}|\mathbf{r})$ only requires the computation of one marginal, i.e., $p(u_l,\mathbf{y}|\mathbf{b})$. Note moreover that the choice of u_l is arbitrary: the algorithm delivers $p(\mathbf{b}|\mathbf{r})$ irrespective of the choice of u_l. The reduced marginalization problem (8.28) may be performed by applying the SP algorithm on the factor graph represented in Fig. 8.2. Note that since the value of $\mathbf{b}$ is fixed, variable node $\mathbf{b}$ no longer appears in the factor graph. Depending on the bit-to-symbol mapping and the error-correcting code considered for the transmission, the factor graph in Fig. 8.2 may not contain cycles. In such a case, marginals $p(u_l,\mathbf{y}|\mathbf{b})$ can be exactly computed and so can be $p(\mathbf{b}|\mathbf{r})$ via (8.29). If the graph associated to (8.28) contains cycles, the SP algorithm only computes an approximation of $p(u_l,\mathbf{y}|\mathbf{b})$ and (8.29) is also an approximation

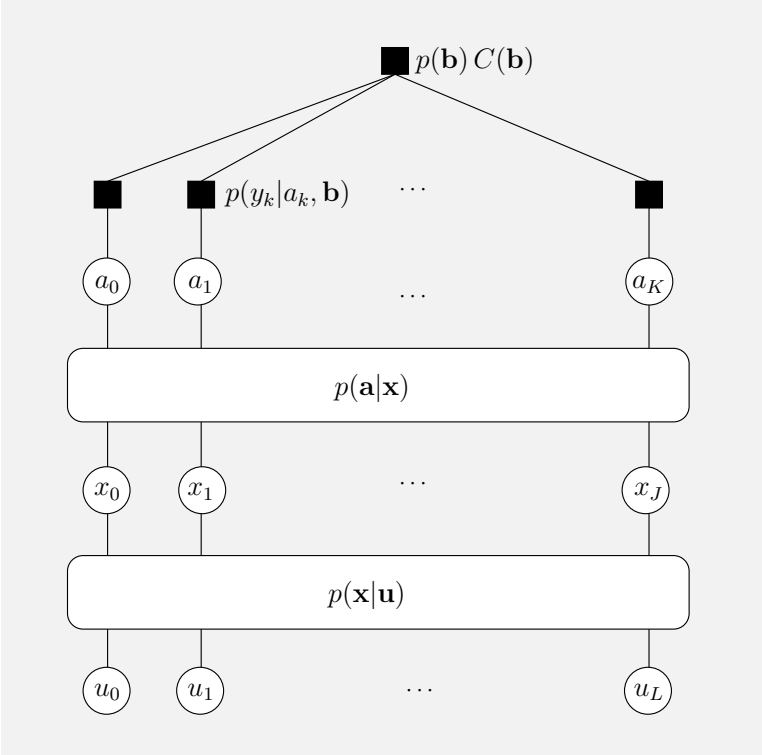

Figure 8.1 Factor-graph for the computation of $p(\mathbf{b}|\mathbf{r})$.

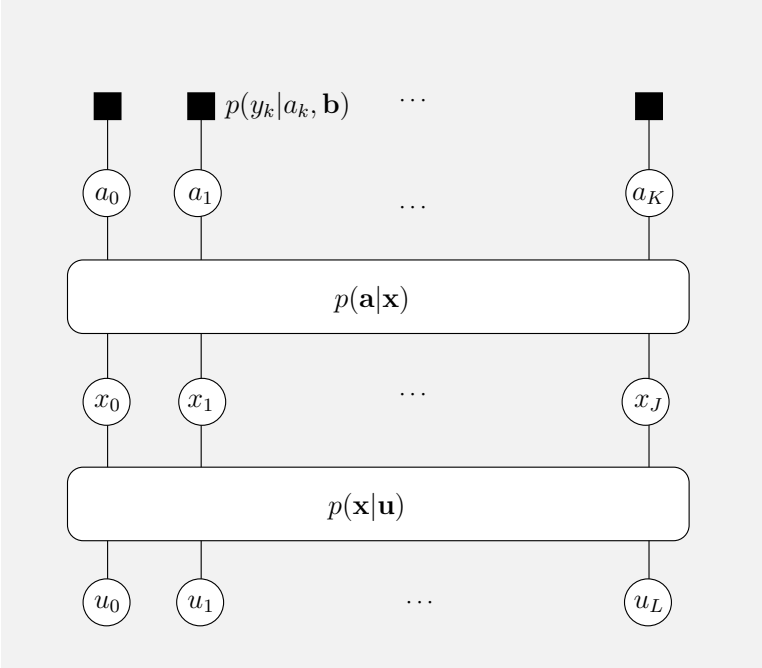

Figure 8.2 Factor-graph for the computation of marginal $p(u_l, \mathbf{r}|\mathbf{b})$.

of $p(\mathbf{b}|\mathbf{r})$.

The factor graph represented in Fig. 8.2 is exactly the same as the one used for the computation of marginals $p(u_l|\mathbf{y},\mathbf{b})$ in chapter 5. Therefore, the SP message-update equations derived in chapter 5 for the computation of $p(u_l|\mathbf{y},\mathbf{b})$ are still valid[6] here. In the case of a BPSK convolutionally-coded transmission for example, the application of the SP algorithm on the factor graph in Fig. 8.2 leads to a BCJR algorithm. For higher-order modulations or for turbo-coded transmissions, the SP algorithm leads to the implementation of a turbo receiver. As a consequence, we see that the computation of $p(\mathbf{b}|\mathbf{r})$ $(\forall\,\mathbf{b} \in \mathcal{B})$ has the same complexity order as computing marginals $p(u_l|\mathbf{y},\mathbf{b})$ under $|\mathcal{B}|$ different hypotheses. Note also that (8.29) could also have been rewritten as

$$p(\mathbf{b}|\mathbf{r}) \propto p(\mathbf{b})\, C(\mathbf{b}) \sum_{x_j} p(x_j,\mathbf{y}|\mathbf{b}), \tag{8.30}$$

$$p(\mathbf{b}|\mathbf{r}) \propto p(\mathbf{b})\, C(\mathbf{b}) \sum_{a_k} p(a_k,\mathbf{y}|\mathbf{b}). \tag{8.31}$$

or

$$p(\mathbf{b}|\mathbf{r}) \propto p(\mathbf{b})\, C(\mathbf{b}) \sum_{t_i} p(t_i,\mathbf{y}|\mathbf{b}), \tag{8.32}$$

where t_i is a "state" variable which may appear in the factorization of the code (see chapter 5). As a consequence, we see that $p(\mathbf{b}|\mathbf{r})$ can be computed by the summation over the "data" variables of any of the marginals available in the factor-graph in Fig. 8.2.

8.2.3 Sequential MAP Synchronization on Observation Subsets

In the previous section, we show how to efficiently compute the MAP estimate which takes into account the observation vector $\mathbf{r}$. In some situations, processing the whole observation vector may however be unnecessary. Indeed, a subset of the available observations may be sufficient to get the required reliability on the decision made about $\mathbf{b}$. In this section we show how to compute a MAP estimate by only considering a subset of the available information. We then emphasize that, depending on the case, the size of the

[6] Note that the messages entering nodes u_l have to be normalized with respect to u_l for the computation of $p(u_l|\mathbf{y},\mathbf{b})$ whereas they have not for the computation of $p(u_l,\mathbf{y}|\mathbf{b})$.

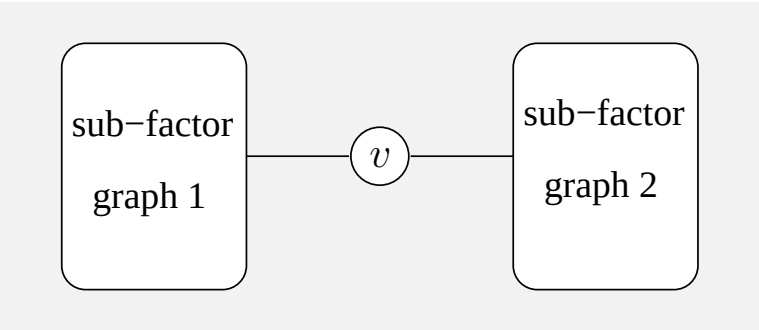

Figure 8.3 Schematic structure of a factor graph enabling to split the operations required by the MAP synchronizer in two parts.

considered subset may be sequentially increased at the expense of a limited number of operations.

Let us assume that the factor graph in Fig. 8.2 may be divided in two sub-factor graphs as in Fig. 8.3. In this figure, v can equivalently represent an information bit u_l, a coded bit x_j, a symbol a_k or a state variable t_i. Let us denote by $\mathbf{u}_1$ (resp. $\mathbf{u_2}$) the subset of the information bits which are present in the sub-factor graph 1 (resp. sub-factor graph 2). Note that, due to the particular structure of the factor-graph, we have $\mathbf{u}_1 \cap \mathbf{u}_2 = \emptyset$. Let us also define $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{a}_1$ and $\mathbf{a}_2$ in the same way. In the sequel, we will assume without loss of generality that $\mathbf{a}_1 = [a_0, a_1, \ldots, a_{K_1-1}]^T$ i.e. is made up with the first K_1 symbols. Based on these assumptions, let us reconsider (8.27):

$$p(\mathbf{b}|\mathbf{r}) \propto p(\mathbf{b})\, C(\mathbf{b}) \sum_{v} \sum_{\sim\{v\}} \prod_{k} p(y_k|a_k, \mathbf{b})\, p(\mathbf{a}|\mathbf{x})\, p(\mathbf{x}|\mathbf{u})\, p(\mathbf{u}), \tag{8.33}$$

$$\begin{aligned} &\propto p(\mathbf{b}) \sum_{v} \Big(C_1(\mathbf{b}) \sum_{\sim\{v\}} \prod_{k=0}^{K_1-1} p(y_k|a_k, \mathbf{b})\, p(\mathbf{a}_1|\mathbf{x}_1)\, p(\mathbf{x}_1|\mathbf{u}_1)\, p(\mathbf{u}_1) \\ &\qquad \times C_2(\mathbf{b}) \sum_{\sim\{v\}} \prod_{k=K_1}^{K} p(y_k|a_k, \mathbf{b})\, p(\mathbf{a}_2|\mathbf{x}_2)\, p(\mathbf{x}_2|\mathbf{u}_2)\, p(\mathbf{u}_2) \Big), \end{aligned} \tag{8.34}$$

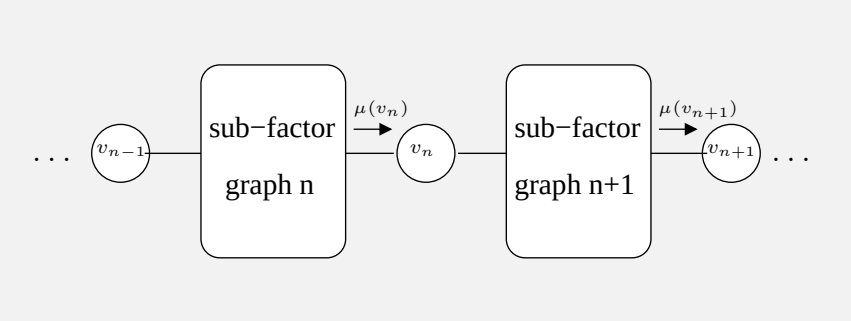

Figure 8.4 Schematic structure of a factor graph enabling sequential MAP estimation.

where $C_1(\mathbf{b})$ and $C_2(\mathbf{b})$ are defined as follows:

$$C_1(\mathbf{b}) = \exp\Big(\frac{1}{2N_0}\sum_{k=0}^{K_1-1}\|y_k(\tau = k_\tau, \nu)\|^2\Big), \tag{8.35}$$

$$C_2(\mathbf{b}) = \exp\Big(\frac{1}{2N_0}\sum_{k=K_1}^{K}\|y_k(\tau = k_\tau, \nu)\|^2\Big). \tag{8.36}$$

If we disregard the message associated to the second sub-factor graph, i.e.

$$C_2(\mathbf{b})\sum_{\sim\{v\}}\prod_{k=K_1}^{K} p(y_k|a_k, \mathbf{b})\, p(\mathbf{a}_2|\mathbf{x}_2)\, p(\mathbf{x}_2|\mathbf{u}_2)\, p(\mathbf{u}_2), \tag{8.37}$$

we actually compute $p(\mathbf{b}|\mathbf{r}_1)$ where $\mathbf{r}_1$ is an observation vector defined as[7]

$$r_1(lT_s) = A\sum_{k=0}^{K_1-1} a_k\, p(lT_s - kT - k_\tau T)\, e^{jk_\theta\Psi} + w(lT_s). \tag{8.38}$$

Indeed, applying formula (8.27) to the computation of $p(\mathbf{b}|\mathbf{r}_1)$, we have

$$p(\mathbf{b}|\mathbf{r}_1) \propto p(\mathbf{b})\, C_1(\mathbf{b}) \sum_{v}\sum_{\sim\{v\}}\prod_{k=0}^{K_1-1} p(y_k|a_k, \mathbf{b})\, p(\mathbf{a}_1|\mathbf{x}_1)\, p(\mathbf{x}_1|\mathbf{u}_1)\, p(\mathbf{u}_1). \tag{8.39}$$

Note that the observation vector $\mathbf{r}_1$ actually contains the samples we would have observed if we had only transmitted the first K_1 symbols. The estimation of $\mathbf{b}$ from $p(\mathbf{b}|\mathbf{r}_1)$ rather than $p(\mathbf{b}|\mathbf{r})$ is therefore equivalent to only take the transmitted power associated the K_1 first symbols into account.

[7] We assume without loss of generality that the frequency phase offset $\nu = 0$.

We can ask the question whether it is possible to progressively increase the power taken into account in the estimation of $\mathbf{b}$ while reusing some intermediate results? This is possible depending on the structure of the factor graph. When the considered factor graph has a structure similar to the one represented in Fig. 8.4, we know from the previous reasoning that $p(\mathbf{b}|\mathbf{r}_n)$ may be computed as follows:

$$p(\mathbf{b}|\mathbf{r}_n) = p(\mathbf{b})\, C_n(\mathbf{b}) \sum_{v_n} \mu_n(v_n), \tag{8.40}$$

where $\mathbf{r}_n$ is the vector of observations which takes into account the transmission of the symbols present in the left-hand part of the factor graph. In a similar way $p(\mathbf{b}|\mathbf{r}_{n+1})$ may be computed as

$$p(\mathbf{b}|\mathbf{r}_{n+1}) = p(\mathbf{b})\, C_{n+1}(\mathbf{b}) \sum_{v_{n+1}} \mu_{n+1}(v_{n+1}). \tag{8.41}$$

Now, message $\mu_{n+1}(v_{n+1})$ required to compute $p(\mathbf{b}|\mathbf{r}_{n+1})$ may be simply computed by applying the SP algorithm on the $(n+1)^{\text{th}}$ factor graph and using message $\mu_n(v_n)$. The operations performed to compute $p(\mathbf{b}|\mathbf{r}_n)$ are therefore not repeated to compute $p(\mathbf{b}|\mathbf{r}_{n+1})$.

8.2.4 Implementation and Complexity of the MAP Synchronizers

In this subsection, we briefly discuss the complexity of the MAP synchronizers described in sections 8.2.1, 8.2.2 and 8.2.3. In particular, we first emphasize that the actual MAP synchronizer (8.1) has surprisingly a complexity at most as large as the one of its suboptimal counterpart (8.17). Then, we show that the complexity may often be significantly decreased by the using the sequential MAP approach described in subsection 8.2.3.

Let us first note that both the actual MAP solution (8.1) and its suboptimal version proposed in (8.17) require the computation of marginal probabilities. In particular, both approaches can be implemented from the knowledge of marginals[8] $p(a_k, \mathbf{y}|\mathbf{b})$. However, the actual MAP solution proposed in section 8.2.2 only requires the knowledge of one marginal whereas suboptimal approach (8.17) implies the computation of all the symbol marginals. This sometimes enables to significantly reduce the complexity in favor of the actual MAP solution. Indeed, let us for instance consider a BPSK convolutionally-coded transmission: marginals $p(a_k, \mathbf{y}|\mathbf{b})$ may be computed by using the SP

[8] $p(a_k|\mathbf{y}, \mathbf{b})$ is equal to $p(a_k, \mathbf{r}|\mathbf{b})$ up to a normalization factor.

message-update equations derived in section 5.4. However, since the MAP synchronizer only requires one single marginal, it has not to compute both the α and β-recursions defined in (5.27)-(5.28): for example, only the β-recursion is necessary to compute $p(a_0, \mathbf{y}|\mathbf{b})$. In this example, the complexity associated to the computation of one marginal is therefore half the complexity required to evaluate all the marginals. This kind of simplification is possible when the factor-graph associated to the computation of $p(a_k, \mathbf{y}|\mathbf{b})$ has no cycles. When the factor graph has cycles, the SP algorithm is iterative and all the messages in the factor graph have to be updated at each iteration. In any case, the computation of the single marginal required by the actual MAP solution is at most as complex as the computation of all the marginals required by suboptimal approach (8.17).

Note also that the computation of the final estimate from marginals $p(a_k, \mathbf{y}|\mathbf{b})$ appears to be slightly easier in the case of the actual MAP solution: we see from (8.31) that the computation of $\hat{\mathbf{b}}_{MAP}$ only require to summarize $p(a_k, \mathbf{y}|\mathbf{b})$ over all possible value of a_k. On the contrary, the computation of the $\mathcal{Q}$-function required in (8.17) implies the computation of the first and second order a posteriori moments of all the transmitted symbols (see section 7.4.1). As a consequence, we can draw the surprising conclusion that the MAP synchronizer (8.1) exhibits a similar or smaller[9] complexity than suboptimal approach (8.17).

Regarding the implementation of the CA MAP synchronizers (8.1) and (8.17), we should insist on the fact that both these estimators require the knowledge of symbol marginals $p(a_k, \mathbf{y}|\mathbf{b})$. Therefore, these synchronizers are only well-suited to receivers based on a CMAP criterion (see chapter 4); otherwise they would require the computation of probabilities $p(a_k, \mathbf{y}|\mathbf{b})$ only for the sake of synchronization and would therefore dramatically increase the complexity. Note that even when a CMAP data decision rule is considered, the implementation of (8.1) or (8.17) implies a quite important increase of the complexity with respect to the complexity of a perfectly-synchronized system. The implementation of (8.17), for example, implies to run $|\mathcal{B}|$ decoding operations. Note that when the factor graph associated to the computation of $p(a_k, \mathbf{y}|\mathbf{b})$ has cycles, the implementation of the MAP synchronizers may be simplified by only running a limited number of SP iterations. In such a case,

[9] depending on the considered case.

the probabilities delivered by the SP algorithm are only approximations of actual probabilities $p(a_k, \mathbf{y}|\mathbf{b})$, and therefore so is the solution computed by the synchronizers.

The sequential MAP estimation proposed in subsection 8.2.3 can enable to avoid a full decoding operation per hypothesis to test. Indeed, we usually only require a limited confidence on the decision made about the synchronization parameters. Therefore, processing the whole frame is not always necessary to ensure a proper synchronization. The sequential MAP method enables to stop the processing when the decision made about $\mathbf{b}$ is reliable enough. At a given step of the sequential MAP, 1-$p(\mathbf{b} = \tilde{\mathbf{b}}|\mathbf{r}_n)$ gives the probability of committing an error if we decide that $\mathbf{b} = \tilde{\mathbf{b}}$. At each step, we have therefore at our disposal a measure of the accuracy of the decision that we can make. The sequential MAP synchronizer can be stopped when a sufficient reliability is achieved.

Let us finally mention that the number of hypotheses to test may be reduced by considering list synchronization: the q most probable candidates are first determined by means of a simple synchronization algorithm; then, only these q candidates are tested by the more complex CA algorithm.

8.3 Data-oriented Synchronization

Error-free data transmission is the ultimate goal of any communication systems. Any algorithm implemented in a receiver should therefore ideally be designed to minimize the probability of committing an error when making a decision about the transmitted sequence $\mathbf{u}$. MAP synchronization optimizes the probability of detection associated to the synchronization parameter but gives no guarantee about the quality of the subsequent decision made about the transmitted data. Surprisingly, the CA synchronizers which have been proposed so far in the literature focus on the MAP criterion but do not consider the optimization of the frame-error rate (FER) or the bit-error rate (BER).

In this section, we take a step back and derive estimators based on the minimization of data-oriented criteria, namely the FER and the BER. Denoting by $p(e_{\mathbf{u}})$ the probability of committing at least one error among the decisions

made about the transmitted bits, we can express the FER as follows:

$$\text{FER} = \int p(e_{\mathbf{u}}|\mathbf{r})\, p(\mathbf{r}) d\mathbf{r}. \tag{8.42}$$

Probability $p(e_{\mathbf{u}}|\mathbf{r})$ may be written as

$$p(e_{\mathbf{u}}|\mathbf{r}) = 1 - p(\mathbf{u} = \hat{\mathbf{u}}(\mathbf{r})|\mathbf{r}), \tag{8.43}$$

where $\hat{\mathbf{u}}(\mathbf{r})$ is the decision made about $\mathbf{u}$ from the observation vector $\mathbf{r}$. On the other hand, the BER achieved by a communication system may be expressed as follows:

$$\text{BER} = \frac{1}{L} \int \Big[\sum_{l=0}^{L} l\, p(e_{\mathbf{u}}^{l}|\mathbf{r}) \Big] p(\mathbf{r})\, d\mathbf{r}, \tag{8.44}$$

where $p(e_{\mathbf{u}}^{l}|\mathbf{r})$ denotes the probability of committing l errors in the decisions made (given $\mathbf{r}$) about the transmitted bits. The BER (8.44) may also be rewritten as

$$\text{BER} = 1 - \frac{1}{L} \int \Big[\sum_{l=1}^{L} p(u_l = \hat{u}_l(\mathbf{r})|\mathbf{r}) \Big] p(\mathbf{r})\, d\mathbf{r}, \tag{8.45}$$

where $\hat{u}_l(\mathbf{r})$ denotes the decision made about u_l.

The remainder of this section is organized as follows. In a first subsection we derive a synchronizer which is optimal in terms of the minimization of the FER. The receiver based on this approach is shown to significatively reduce the complexity with respect to the FER-wise optimal receiver (see (4.2)). In the second subsection, we consider the equations of the BER-wise optimal receiver, taking into account that the synchronization parameters are discrete. Then, in a last subsection, we derive a decision rule on the synchronization parameters which is optimal in terms of the minimization of the BER. Unlike the FER-wise approach, the synchronizer optimizing the BER does not reduce the complexity of implementation with respect to the optimal solution (4.4) and is therefore shown to be useless.

8.3.1 FER-wise Optimal Synchronization

In this section, we derive an estimator of $\mathbf{b}$ which is based on the minimization of the frame-error rate (FER). The proposed estimator is shown to have a complexity comparable to other CA discrete-parameter estimators: it requires one decoding operation per hypothesis to test. It is moreover shown to be

easily implementable in any kind of receiver, unlike most CA synchronizers proposed in the literature. When the CMAP decision rule is used to make a decision on the transmitted sequence (see chapter 4), we moreover emphasize that the proposed synchronizer combined with the CMAP decision rule leads, in a good approximation, to a joint ML problem.

From (8.42) and (8.43) it is clear that the FER-wise optimal decision on $\mathbf{u}$ is given by

$$\hat{\mathbf{u}}_{MAP} = \arg\max_{\tilde{\mathbf{u}} \in \mathcal{U}} \; p(\mathbf{u} = \tilde{\mathbf{u}}|\mathbf{r}). \tag{8.46}$$

Unfortunately, the latter maximization problem does usually not have any simple analytical solution: in the general case, we are required to perform $|\mathcal{U}|$ evaluations of the objective function $p(\mathbf{u}|\mathbf{r})$ in order to find its maximum. Therefore, rather than implementing this quite complicated solution, CMAP sequence estimation is often preferred (see chapter 4), i.e.

$$\hat{\mathbf{u}}_{CMAP} = \arg\max_{\tilde{\mathbf{u}} \in \mathcal{U}} \; p(\mathbf{u} = \tilde{\mathbf{u}}|\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}), \tag{8.47}$$

where $\hat{\mathbf{b}}$ is an estimate of discrete synchronization parameter $\mathbf{b}$. In fact, the CMAP sequence estimator may be regarded as a particular case of a general family of sequence estimation algorithms operating as follows: they first compute an estimate $\hat{\mathbf{b}}$ of $\mathbf{b}$; then, they perform data detection assuming that $\mathbf{b}$ is equal to $\hat{\mathbf{b}}$, i.e.

$$\hat{\mathbf{u}} = f(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}), \tag{8.48}$$

where $f(\cdot)$ denotes some sequence decision rule. In the sequel, we will use the short-hand notation $f(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}) = f(\mathbf{r}, \hat{\mathbf{b}})$.

From (8.48), it is clear that the estimate of the transmitted sequence depends on the value of the estimate $\hat{\mathbf{b}}$ made about $\mathbf{b}$. In this context, we may ask the question of the choice of $\hat{\mathbf{b}}$ which would minimize the probability of making a wrong decision about $\mathbf{u}$, i.e. minimize the FER. Using (8.42) and (8.43), the answer to this question is simply given by

$$\hat{\mathbf{b}}_{FER} = \arg\max_{\hat{\mathbf{b}} \in \mathcal{B}} p(\mathbf{u} = f(\mathbf{r}, \hat{\mathbf{b}})|\mathbf{r}). \tag{8.49}$$

The determination of the FER-wise optimal synchronization estimate $\hat{\mathbf{b}}_{FER}$ may therefore be done as follows. First, a sequence estimate has to be

computed according to (8.48) for each possible value of $\hat{\mathbf{b}} \in \mathcal{B}$. Probability $p(\mathbf{u} = \hat{\mathbf{u}}|\mathbf{r})$ is then computed for each sequence estimate. The FER-wise optimal estimate $\hat{\mathbf{b}}_{FER}$ is finally set as the one leading to the sequence estimate $\hat{\mathbf{u}} = f(\mathbf{r}, \hat{\mathbf{b}}_{FER})$ which maximizes $p(\mathbf{u}|\mathbf{r})$. Notice that $1 - p(\mathbf{u} = \hat{\mathbf{u}}|\mathbf{r})$ is the probability of committing an error if we decide that sequence $\hat{\mathbf{u}}$ has been transmitted. Therefore, decision rule (8.49) is implicitly equivalent to choosing, among $|\mathcal{B}|$ possible sequences, the one which minimizes the probability of wrong detection.

Unlike optimal solution (8.46) which requires $|\mathcal{U}|$ evaluations of $p(\mathbf{u}|\mathbf{r})$, maximization problem (8.49) only implies $|\mathcal{B}|$ such evaluations. Since $|\mathcal{B}| \ll |\mathcal{U}|$, the approach based on (8.48) and (8.49) is therefore much less complex than optimal solution (8.46). Note that since $\mathbf{b}$ is a discrete-valued parameter, $p(\mathbf{u}|\mathbf{r})$ is easy to evaluate. Indeed,

$$\begin{aligned} p(\mathbf{u}|\mathbf{r}) &= \sum_{\mathbf{b}} p(\mathbf{u}|\mathbf{r}, \mathbf{b})\, p(\mathbf{b}|\mathbf{r}) \\ &\propto \sum_{\mathbf{b}} p(\mathbf{r}|\mathbf{u}, \mathbf{b})\, p(\mathbf{u})\, p(\mathbf{b}), \end{aligned} \tag{8.50}$$

where $\propto$ denotes equality up to a normalization factor. $p(\mathbf{u})$ and $p(\mathbf{b})$ are usually uniform whereas $p(\mathbf{r}|\mathbf{u}, \mathbf{b})$ is a Gaussian distribution. Since $|\mathcal{B}|$ is small, the number of terms in (8.50) is also small and the evaluation of (8.50) is straightforward. In fact, we may note that, unlike the MAP synchronizers derived in section 8.2, the FER-wise optimal synchronizer (8.49) does not require the computation of marginals $p(a_k, \mathbf{r}|\mathbf{b})$. Instead, it bases its estimation on the rule used to make a decision about the transmitted sequence, i.e. $f(\cdot)$. The FER-wise optimal synchronizer (8.49) may therefore be easily implemented in any kind of receiver.

We now consider the particular case where the sequence decision rule $f(\mathbf{r}, \hat{\mathbf{b}})$ is the CMAP sequence estimator (8.47). We will show that the FER-oriented optimal synchronization (8.49) combined with CMAP sequence estimation (8.47) reduces in good approximation to a joint maximum-likelihood problem, i.e.

$$(\hat{\mathbf{u}}_{CMAP}, \hat{\mathbf{b}}_{FER}) \simeq \operatorname{argmax}_{(\tilde{\mathbf{u}}, \tilde{\mathbf{b}}) \in \mathcal{U} \times \mathcal{B}}\ p(\mathbf{r}|\mathbf{u} = \tilde{\mathbf{u}}, \mathbf{b} = \tilde{\mathbf{b}}). \tag{8.51}$$

Indeed, using (8.49) with CMAP sequence decision rule, the search of the synchronization parameter value $\hat{\mathbf{b}}_{FER}$ minimizing the sequence error probability

becomes

$$\hat{\mathbf{b}}_{FER} = \arg\max_{\hat{\mathbf{b}} \in \mathcal{B}} \; p\big(\mathbf{u} = \hat{\mathbf{u}}_{CMAP}(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}) | \mathbf{r}\big), \tag{8.52}$$

where the notation $\hat{\mathbf{u}}_{CMAP}(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}})$ recalls the fact that the decision made about the transmitted sequence both depends on $\mathbf{r}$ and $\hat{\mathbf{b}}$. Expanding then $p(\mathbf{u}|\mathbf{r})$ as

$$p(\mathbf{u}|\mathbf{r}) = \sum_{\mathbf{b}} p(\mathbf{u}|\mathbf{r}, \mathbf{b})\; p(\mathbf{b}|\mathbf{r}), \tag{8.53}$$

and taking into account that

$$p(\mathbf{u}|\mathbf{b}, \mathbf{r})\; p(\mathbf{b}|\mathbf{r}) = \frac{p(\mathbf{r}|\mathbf{u}, \mathbf{b}) p(\mathbf{u})\; p(\mathbf{b})}{p(\mathbf{r})}, \tag{8.54}$$

we get by dropping the terms independent[10] of $\mathbf{b}$:

$$\hat{\mathbf{b}}_{FER} = \arg\max_{\hat{\mathbf{b}} \in \mathcal{B}} \Big\{ p(\mathbf{r}|\mathbf{u} = \hat{\mathbf{u}}_{CMAP}(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}), \mathbf{b} = \hat{\mathbf{b}}) + \sum_{\tilde{\mathbf{b}} \neq \hat{\mathbf{b}}} p(\mathbf{r}|\mathbf{u} = \hat{\mathbf{u}}_{CMAP}(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}), \mathbf{b} = \tilde{\mathbf{b}}) \Big\}.$$

In the same way, using (8.54) and dropping the terms independent of $\mathbf{u}$, the CMAP sequence decision rule (8.47) may be written as

$$\hat{\mathbf{u}}_{CMAP}(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}) = \arg\max_{\tilde{\mathbf{u}} \in \mathcal{U}} \; p(\mathbf{r}|\mathbf{u} = \tilde{\mathbf{u}}, \mathbf{b} = \hat{\mathbf{b}}). \tag{8.55}$$

Note that, due to the Gaussian nature of the noise, the CMAP decision rule (8.55) actually reduces to choosing the information sequence leading to the symbol sequence[11] $\mathbf{a}(\mathbf{u})$ which minimizes the Euclidean distance with observation vector $\mathbf{r}$ under the assumption $\mathbf{b} = \hat{\mathbf{b}}$.

In the sequel, we make the assumption that $\forall\; \tilde{\mathbf{b}} \neq \hat{\mathbf{b}}$ we have

$$p(\mathbf{r}|\mathbf{u} = \hat{\mathbf{u}}_{CMAP}(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}), \mathbf{b} = \hat{\mathbf{b}}) \gg p(\mathbf{r}|\mathbf{u} = \hat{\mathbf{u}}_{CMAP}(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}), \mathbf{b} = \tilde{\mathbf{b}}) \tag{8.56}$$

i.e. we assume the Euclidean distance between observation vector $\mathbf{r}$ and CMAP sequence decision $\mathbf{a}\Big(\hat{\mathbf{u}}_{CMAP}(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}})\Big)$ under the assumption $\mathbf{b} \neq \hat{\mathbf{b}}$ is large. Note that this approximation usually makes sense in the context

[10] We assume that $p(\mathbf{b})$ is uniform.

[11] $\mathbf{a}(\mathbf{u})$ denotes the symbol sequence obtained by coding and mapping information sequence $\mathbf{u}$.

of synchronization. For the sake of illustration, let us consider the case of phase ambiguity resolution for a BSPK transmission. As mentioned above, the CMAP sequence decoder delivers the information sequence $\hat{\mathbf{u}}_{CMAP}(\mathbf{r},\hat{\mathbf{b}})$ whose corresponding coded symbol sequence $\mathbf{a}\Big(\hat{\mathbf{u}}_{CMAP}(\mathbf{r},\hat{\mathbf{b}})\Big)$ has the smallest Euclidean distance with the received observation vector $\mathbf{r}$ under the assumption $\mathbf{b}=\hat{\mathbf{b}}$. Considering the probability of $\mathbf{r}$ given $\mathbf{a}\Big(\hat{\mathbf{u}}_{CMAP}(\mathbf{r},\hat{\mathbf{b}})\Big)$ under the complementary assumption, i.e. $\mathbf{b}=\hat{\mathbf{b}}\,e^{j\pi}$, we have

$$\begin{aligned} p(\mathbf{r}|\mathbf{u}=\hat{\mathbf{u}}_{CMAP}(\mathbf{r},\hat{\mathbf{b}}),\mathbf{b}=\hat{\mathbf{b}}\,e^{j\pi}) &= p(\mathbf{r}|\mathbf{a}=\mathbf{a}\big(\hat{\mathbf{u}}_{CMAP}(\mathbf{r},\hat{\mathbf{b}})\big),\mathbf{b}=\hat{\mathbf{b}}\,e^{j\pi}) \\ &= p(\mathbf{r}|\mathbf{a}=\mathbf{a}\big(\hat{\mathbf{u}}_{CMAP}(\mathbf{r},\hat{\mathbf{b}})\big)\,e^{j\pi},\mathbf{b}=\hat{\mathbf{b}}), \end{aligned} \tag{8.57}$$

where (8.57) directly derives from the particular structure of the received signal (2.8). Now, $\mathbf{a}\big(\hat{\mathbf{u}}_{CMAP}(\mathbf{r},\hat{\mathbf{b}})\big)\,e^{j\pi}$ is the sequence which has the largest Hamming distance with respect to sequence $\mathbf{a}\big(\hat{\mathbf{u}}_{CMAP}(\mathbf{r},\hat{\mathbf{b}})\big)$. Since $\mathbf{a}\big(\hat{\mathbf{u}}_{CMAP}(\mathbf{r},\hat{\mathbf{b}})\big)$ is close to $\mathbf{r}$ in terms of Euclidean distance, $\mathbf{a}\big(\hat{\mathbf{u}}_{CMAP}(\mathbf{r},\hat{\mathbf{b}})\big)\,e^{j\pi}$ is proportionally quite far. This implies (8.56).

Taking (8.56) into account, we have

$$\hat{\mathbf{b}}_{FER} \simeq \arg\max_{\hat{\mathbf{b}}\in\mathcal{B}}\; p(\mathbf{r}|\mathbf{u}=\hat{\mathbf{u}}_{CMAP}(\mathbf{r},\mathbf{b}=\hat{\mathbf{b}}),\mathbf{b}=\hat{\mathbf{b}}). \tag{8.58}$$

Combining (8.55) and (8.58), we get (8.51). As a conclusion, considering a CMAP sequence decision rule (8.47), we have under assumption (8.56) that the FER-wise optimal decision on $\mathbf{u}$ and $\mathbf{b}$ is the solution of the joint maximum-likelihood problem (8.51).

8.3.2 The BER-wise Optimal Receiver Revisited

From (8.45), it is clear that the BER-wise optimal bit estimator is given by

$$\hat{u}_{MAP,l} = \arg\max_{\tilde{u}_l\in\{0,1\}}\; p(u_l=\tilde{u}_l|\mathbf{r}). \tag{8.59}$$

As mentioned in chapter 4, this maximization problem is usually intractable since the computation of $p(u_l|\mathbf{r})$ requires a multidimensional integral over the synchronization parameter space. If the synchronization parameters are discrete-valued, this integral reduces to a finite sum and, as we will see, the evaluation of (8.59) may be done with a tractable complexity.

Let us focus on the computation of $p(u_l|\mathbf{r})$. The computation of this marginal

has been placed in the context of the factor graph representation and the SP algorithm in [79]. The authors actually applied the SP algorithm on the factor graph of the overall joint probability $p(\mathbf{u}, \mathbf{r}, \mathbf{b})$, i.e.,

$$p(u_l|\mathbf{r}) \propto \sum_{\sim\{u_l\}} p(\mathbf{u}, \mathbf{r}, \mathbf{b}). \tag{8.60}$$

Unfortunately, due to the presence of cycles in the graph of $p(\mathbf{u}, \mathbf{r}, \mathbf{b})$, the proposed approach turned out to lead to poor results. In this section, we show that the computation of $p(u_l|\mathbf{r})$ may be performed exactly by applying the SP in a slightly different way.

Let us expand $p(u_l|\mathbf{r})$ as

$$\begin{aligned} p(u_l|\mathbf{r}) &\propto \sum_{\mathbf{b}\in\mathcal{B}} \sum_{\mathbf{u}\setminus\{u_l\}} p(\mathbf{u}, \mathbf{r}, \mathbf{b}), \\ &\propto \sum_{\mathbf{b}\in\mathcal{B}} \left(\sum_{\mathbf{u}\setminus\{u_l\}} p(\mathbf{u}, \mathbf{r}|\mathbf{b}) \right) p(\mathbf{b}). \end{aligned} \tag{8.61}$$

First, let us notice that the summation over $\mathbf{b}$ contains a small number of terms and is therefore easy to perform. On the contrary, marginalization $p(u_l, \mathbf{r}|\mathbf{b}) = \sum_{\mathbf{u}\setminus\{u_l\}} p(\mathbf{u}, \mathbf{r}|\mathbf{b})$ might be more complex and should be considered carefully. Now, this marginalization problem may be solved efficiently by using the factor-graph representation and the SP algorithm (see chapter 5). The problem of computing marginals $p(u_l, \mathbf{r}|\mathbf{b})$ has already been discussed in section 8.2.2. In particular, we show that the same factor graph and therefore the same SP message update rules may be used for the computation of $p(u_l, \mathbf{r}|\mathbf{b})$ and $p(u_l|\mathbf{r}, \mathbf{b})$.

Note that the factor graph associated to $p(u_l, \mathbf{r}|\mathbf{b})$ does not have necessarily cycles whereas the one associated to $p(u_l, \mathbf{r}, \mathbf{b})$ always have cycles when considering coded transmissions (see [79]). In fact, splitting the overall summation $\sum_{\sim\{u_l\}}$ in (8.60) in two summations as in (8.61) enables to apply the SP algorithm on smaller factor graphs, possibly without cycles. As a consequence, the proposed application of the SP algorithm enables to efficiently compute the actual optimal BER-wise solution when the factor graph associated to $p(u_l, \mathbf{r}|\mathbf{b})$ is cycle-free. As a consequence, since $p(u_l|\mathbf{r})$ may be computed by a simple weighted sum of probabilities $p(u_l, \mathbf{r}|\mathbf{b})$, the complexity associated to the implementation of the bit-wise optimal receiver is $|\mathcal{B}|$ times larger than the complexity of a CMAP bit estimator (4.4). This complexity

remains tractable and the BER-wise optimal solution may therefore be considered in practice.

8.3.3 The BER-wise Optimal Synchronizer

In this section, we derive a estimator of $\mathbf{b}$ optimal in the sense of the minimization of the BER. We emphasize that its implementation complexity is similar to the implementation of the BER-wise optimal receiver, making its practical implementation useless.

Let us assume that the decisions on the transmitted bits u_l are made as follows: first an estimate $\hat{\mathbf{b}}$ of $\mathbf{b}$ is computed; then, a decision on u_l is made according to some decision rule f_l by assuming that $\mathbf{b}$ is known and equal to $\hat{\mathbf{b}}$, i.e.

$$\hat{u}_l = f_l(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}). \tag{8.62}$$

In the sequel, we will use the following short-hand notation: $f_l(\mathbf{r}, \mathbf{b} = \hat{\mathbf{b}}) = f_l(\mathbf{r}, \hat{\mathbf{b}})$. From (8.62) it is clear that the decision made about u_l depends on $\hat{\mathbf{b}}$. In this context we may ask the question of the estimate $\hat{\mathbf{b}}$ which minimizes the probability of committing an error on the decision made on u_l. Using (8.45) and (8.62), the solution to this problem, say $\hat{\mathbf{b}}_{BER}$, is given by

$$\hat{\mathbf{b}}_{BER} = \arg\max_{\hat{\mathbf{b}} \in \mathcal{B}} \left\{ \sum_{l=0}^{L-1} p(u_l = f_l(\mathbf{r}, \hat{\mathbf{b}}) | \mathbf{r}) \right\}. \tag{8.63}$$

The BER-wise optimal estimate $\hat{\mathbf{b}}_{BER}$ can therefore be obtained as follows. For each possible value of $\hat{\mathbf{b}} \in \mathcal{B}$, we have to compute $\hat{u}_l = f_l(\mathbf{r}, \hat{\mathbf{b}})$ and evaluate cost function $\sum_{l=0}^{L-1} p(u_l = f_l(\mathbf{r}, \hat{\mathbf{b}}) | \mathbf{r})$. Then, we set $\hat{\mathbf{b}}_{BER}$ to the value of $\hat{\mathbf{b}} \in \mathcal{B}$ which maximizes the cost function.

The computation of marginals $p(u_l|\mathbf{r})$ appearing in (8.63) dominates the complexity of this approach. Now, when the latter marginals are at our disposal, the BER-wise optimal solution may be implemented without significant additional complexity. The complexity of the approach based on a BER-wise optimal synchronizer is therefore the same as the complexity of the BER-wise optimal receiver.

8.4 Main Points

In this chapter, we derive code-aided (CA) synchronization algorithms for the estimation of discrete-valued parameters:

- The estimators are derived according to two different criteria: we consider either the minimization of the probability of wrong decision on the synchronization parameters or the minimization of the probability of wrong decision on the transmitted data.
- The estimator minimizing the probability of wrong decision on the synchronization parameters is the MAP estimator. The MAP synchronizer may be (exactly or approximately) implemented via the SP algorithm and the factor-graph representation (see section 8.2.2). In particular, it requires the computation of marginals $p(a_k, \mathbf{r}|\mathbf{b})$. Its complexity is tractable and corresponds, at most, to $|\mathcal{B}|$ times the complexity of a perfectly synchronized CMAP bit detector, where $|\mathcal{B}|$ is the number of possible values that can be taken by the synchronization parameters.
- A suboptimal version of the MAP synchronizer is proposed in an ad-hoc fashion in [18] and justified in this chapter (see section 8.2.1). It is emphasized that this synchronizer may be thought as a "data-detection-oriented" approximation of the actual MAP synchronizer: the synchronizer is only required to compute the exact MAP estimate when the probability of subsequently making the right sequence decision is large.
- We then consider the design of synchronizers which are optimal in the sense of the minimization of the probability of wrong decision about the transmitted data. Synchronizers based on the minimization of the frame-error rate (FER) and the bit-error rate (BER) are considered. Unlike MAP synchronizers, the FER-wise and BER-wise optimal synchronizers infer the value of the synchronization parameters based on the rule used to make a decision about the data.
- The FER-wise optimal synchronizer only implies the evaluation of a Gaussian distribution and is therefore easily implementable in any kind of receivers.
- The BER-wise optimal synchronizer is shown to have the same complexity as the BER-wise optimal receiver (4.4) and is therefore useless in practice.

8.5 Appendices

8.5.1 Proof of (8.9)

In this appendix, we show that (8.9) holds. Let us consider the probability of making a wrong decision on the transmitted sequence i.e.

$$\begin{aligned} p(e_{\mathbf{u}}|\mathbf{b}=\mathbf{b}^\star) &= \sum_{\tilde{\mathbf{b}}\in\mathcal{B}} \int p(e_{\mathbf{u}}, \mathbf{r}, \hat{\mathbf{b}}=\tilde{\mathbf{b}}|\mathbf{b}=\mathbf{b}^\star)\, d\mathbf{r} \\ &= \sum_{\tilde{\mathbf{b}}\in\mathcal{B}} \int p(e_{\mathbf{u}}|\mathbf{r}, \mathbf{b}=\mathbf{b}^\star, \hat{\mathbf{b}}=\tilde{\mathbf{b}}) p(\hat{\mathbf{b}}=\tilde{\mathbf{b}}|\mathbf{r}, \mathbf{b}=\mathbf{b}^\star)\, p(\mathbf{r}|\mathbf{b}=\mathbf{b}^\star)\, d\mathbf{r}. \end{aligned}$$

Note that $\hat{\mathbf{b}}$ is a univocal function of the observation vector $\mathbf{r}$, i.e. $\hat{\mathbf{b}} = f(\mathbf{r})$. Denoting by $\mathbb{I}\{\cdot\}$ the indicator function i.e. the function equal to 1 if the statement between the braces is true and 0 otherwise, we have therefore

$$p(\hat{\mathbf{b}}=\tilde{\mathbf{b}}|\mathbf{r}, \mathbf{b}=\mathbf{b}^\star) = \mathbb{I}\{\tilde{\mathbf{b}} = f(\mathbf{r})\}. \tag{8.64}$$

As a consequence, the probability of wrong detection may be rewritten as

$$p(e_{\mathbf{u}}|\mathbf{b}=\mathbf{b}^\star) = \int p(e_{\mathbf{u}}|\mathbf{r}, \mathbf{b}=\mathbf{b}^\star, \hat{\mathbf{b}}=f(\mathbf{r}))\, p(\mathbf{r}|\mathbf{b}=\mathbf{b}^\star)\, d\mathbf{r}. \tag{8.65}$$

Let us now consider the subset of observation vectors, say $\mathcal{R}_{\mathbf{b}}$ (resp. $\bar{\mathcal{R}}_{\mathbf{b}}$), for which the decision rule $f(\mathbf{r})$ leads to a correct (resp. wrong) decision on $\mathbf{b}$. The integral in (8.65) may then be broken into two parts:

$$\begin{aligned} p(e_{\mathbf{u}}|\mathbf{b}=\mathbf{b}^\star) &= \int_{\mathcal{R}_{\mathbf{b}}} p(e_{\mathbf{u}}|\mathbf{r}, \mathbf{b}=\mathbf{b}^\star, \hat{\mathbf{b}}=\mathbf{b}^\star)\, p(\mathbf{r}|\mathbf{b}=\mathbf{b}^\star)\, d\mathbf{r} \\ &\quad + \int_{\bar{\mathcal{R}}_{\mathbf{b}}} p(e_{\mathbf{u}}|\mathbf{r}, \mathbf{b}=\mathbf{b}^\star, \hat{\mathbf{b}}\neq\mathbf{b}^\star)\, p(\mathbf{r}|\mathbf{b}=\mathbf{b}^\star)\, d\mathbf{r}. \end{aligned} \tag{8.66}$$

On the one hand, the probability of making a wrong decision $\hat{\mathbf{a}}$ on the transmitted sequence whereas we have correctly estimated $\mathbf{b}^\star$ is given by

$$p(e_{\mathbf{u}}|\mathbf{r}, \mathbf{b}=\mathbf{b}^\star, \hat{\mathbf{b}}=\mathbf{b}^\star) = 1 - \max_{\tilde{\mathbf{u}}\in\mathcal{U}} \{p(\mathbf{u}=\tilde{\mathbf{u}}\,|\,\mathbf{r}, \mathbf{b}=\mathbf{b}^\star)\}, \tag{8.67}$$

where we have used the fact that we consider a CMAP sequence decision rule (4.12). Since there is a bijective relation between the information-bit sequence $\mathbf{u}$ and the symbol sequence $\mathbf{a}$, (8.67) may also be rewritten as

$$p(e_{\mathbf{u}}|\mathbf{r}, \mathbf{b}=\mathbf{b}^\star, \hat{\mathbf{b}}=\mathbf{b}^\star) = 1 - \max_{\tilde{\mathbf{a}}\in\mathcal{A}^K} \{p(\mathbf{a}=\tilde{\mathbf{a}}\,|\,\mathbf{r}, \mathbf{b}=\mathbf{b}^\star)\}. \tag{8.68}$$

On the other hand, a wrong decision on $\mathbf{b}$ implies with very high probability a wrong decision on the transmitted sequence i.e.

$$p(e_{\mathbf{u}}|\mathbf{r}, \mathbf{b}=\mathbf{b}^\star, \hat{\mathbf{b}}\neq\mathbf{b}^\star) \simeq 1. \tag{8.69}$$

Taking (8.68) and (8.69) into account, we have

$$p(e_{\mathbf{a}}|\mathbf{b}=\mathbf{b}^{\star}) = \int_{\mathcal{R}_{\mathbf{b}}} \left(1 - \max_{\tilde{\mathbf{a}}\in\mathcal{A}^K} p(\mathbf{a}=\tilde{\mathbf{a}}|\mathbf{r},\mathbf{b}=\mathbf{b}^{\star})\right) p(\mathbf{r}|\mathbf{b}=\mathbf{b}^{\star})d\mathbf{r} + \int_{\bar{\mathcal{R}}_{\mathbf{b}}} p(\mathbf{r}|\mathbf{b}=\mathbf{b}^{\star})d\mathbf{r}. \tag{8.70}$$

Since

$$\int_{\mathcal{R}_{\mathbf{b}}} p(\mathbf{r}|\mathbf{b}=\mathbf{b}^{\star})d\mathbf{r} + \int_{\bar{\mathcal{R}}_{\mathbf{b}}} p(\mathbf{r}|\mathbf{b}=\mathbf{b}^{\star})d\mathbf{r} = 1, \tag{8.71}$$

we finally get (8.9).

Lower Bounds on the Synchronizer Performance

9

In this chapter, we derive lower bounds on the performance of the SDD synchronizers presented in chapter 7. We first propose two lower bounds on the MSE achievable by a SDD synchronizer. Then, we derive a lower bound on the speed of convergence of the EM synchronizer derived in section 7.4.1.

9.1 Lower Bounds on the MSE

A natural question which arises when designing synchronizers is the ultimate accuracy which may be achieved in the synchronization operation. In order to answer this question, lower bounds on the achievable accuracy have been developed. In particular, the Cramer-Rao bound (CRB) [35], which is a lower bound on the variance of any unbiased estimator, is usually used as a benchmark for assessing the performance of practical estimators. The CRB associated to the estimation of synchronization parameters has already been considered in a number of papers. We mention in particular the following contributions [80], [81], [13], [12], [82]. In [80] and [81], the CRB related to the carrier phase estimation is derived for different symbol constellations assuming uncoded transmission. The Cramer-Rao bound associated to the timing estimation is considered for uncoded transmissions in [13]. In [12], the carrier phase CRB for a coded transmission is computed. Finally, the CRB related to the joint estimation of the carrier phase and the timing epoch is computed in [82]. In practice the modified CRB (MCRB) [42] is also often considered. Indeed, this bound is much simpler to evaluate than the CRB and, although in general looser than the CRB, provides a good benchmark for assessing synchronizer performance at medium-to-high SNR [44].

Recently, iterative synchronizers handling the soft information available in (turbo) receivers have been proposed (see chapter 7). Although usually leading to very good performance after a large number of iterations, the question of the optimality of these synchronizers at a given iteration has not yet been answered. In this section, we derive "soft-information-aware" lower bounds on the synchronization mean square error (MSE). In other words, we intend to answer the following question: *"Given an amount of soft information about the transmitted sequence, what is the best performance achievable by any (unbiased) soft-information-directed estimator ?"*

The section is organized as follows. Firstly, the expression of the CRB taking into account a deterministic symbol soft-information vector is derived. Then, this bound is extended to the case of a random soft-information vector. This bound is referred to as soft CRB (SCRB). Another lower bound, easier to evaluate than the SCRB, is also proposed: the soft modified CRB (SMCRB). The properties of the SCRB and the SMCRB are derived and discussed. In a last part, we focus on the practical evaluation of the SCRB and the SMCRB.

9.1.1 Cramer-Rao Bound in the Presence of a Deterministic Soft-information Vector

As mentioned in the introduction, soft-information-directed synchronizers have recently appeared in the literature (see chapter 7). These synchronizers draw some information about the transmitted symbols from a vector of soft information, say $\mathbf{e} = [e_0, e_1 \dots e_{K-1}]^T$. It is usually assumed that the vector of soft information is an independent source of information i.e. $\mathbf{e}$ is independent of $\mathbf{b}$ and $\mathbf{r}$. Moreover, it is also assumed that each element e_k only contains information about a_k. The vector of soft information may therefore be characterized by the distribution:

$$p(\mathbf{e}|\mathbf{a}) = \prod_k p(e_k|a_k). \tag{9.1}$$

In this subsection we consider the expression of the CRB when such a vector of symbol soft information is available. Note that the above assumptions are usually not verified in practice: the soft information vector $\mathbf{e}$ handled by the receiver is a function of $\mathbf{r}$ and is therefore not independent of it. However, we will see that the bounds derived under these assumptions give interesting

insights into the behavior of SDD synchronizers.

The statistical model taking into account a soft information vector $\mathbf{e}$ and accounting for the distribution of the observations may be written as

$$p(\mathbf{r}|\mathbf{b},\mathbf{e}) = \sum_{\mathbf{a}\in\mathcal{A}^K} p(\mathbf{r}|\mathbf{a},\mathbf{e},\mathbf{b})\, p(\mathbf{a}|\mathbf{e}), \tag{9.2}$$

$$= \sum_{\mathbf{a}\in\mathcal{A}^K} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{a}|\mathbf{e}), \tag{9.3}$$

where (9.3) follows from the fact that $\mathbf{r}$ is independent of soft-information vector $\mathbf{e}$. Using the standard expression of the Fisher information matrix defined in (3.41), we may write

$$\mathbf{F}(\mathbf{b},\mathbf{e}) = E_{\mathbf{r}|\mathbf{b},\mathbf{e}}[\nabla \log p(\mathbf{r}|\mathbf{b},\mathbf{e})\, \nabla^H \log p(\mathbf{r}|\mathbf{b},\mathbf{e})], \tag{9.4}$$

and the CRB associated to the estimation of $\mathbf{b}$ given $\mathbf{e}$ is given by

$$\mathrm{CRB}(\mathbf{b},\mathbf{e}) = \sum_i \mathbf{F}_{ii}^{-1}(\mathbf{b},\mathbf{e}). \tag{9.5}$$

From section 7.5.1, it is straightforward to show that factors $\nabla \log p(\mathbf{r}|\mathbf{b},\mathbf{e})$, required to evaluate the Fisher information matrix, may be computed as

$$\nabla \log p(\mathbf{r}|\mathbf{b},\mathbf{e}) = E_{\mathbf{a}|\mathbf{r},\mathbf{b},\mathbf{e}}[\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b})]. \tag{9.6}$$

This expression enables to efficiently evaluate the Fisher information matrix. Indeed, on the one hand we see from (7.43)-(7.46) that the element of the gradient $\nabla \log p(\mathbf{r}|\mathbf{b},\mathbf{e})$ may be straightforwardly computed from the knowledge of marginals $p(a_k|\mathbf{r},\mathbf{b})$. On the other hand, the remaining expectation with respect to $\mathbf{r}$ given $\mathbf{b}$, required to evaluate (9.4), may be well-approximated by an arithmetical average, that is to say by computing and averaging factors $\nabla \log p(\mathbf{r}|\mathbf{b},\mathbf{e})\, \nabla^H \log p(\mathbf{r}|\mathbf{b},\mathbf{e})$ for a large number of independently-generated vectors $\mathbf{r}$.

9.1.2 Lower Bounds on the MSE in the Presence of a Random Soft-information Vector

In the previous subsection, we show how to efficiently compute the Cramer-Rao bound associated to $\mathbf{b}$ for a particular realization of the soft-information vector $\mathbf{e}$. In practice however, one would like to evaluate the synchronizer performance given a soft-information distribution $p(\mathbf{e})$. In this section, we

consider lower bounds on the MSE for a random soft-information vector.

Since for any soft-information vector $\mathbf{e}$, we have that[1]

$$E_{\mathbf{r}|\mathbf{b},\mathbf{e}}\left[\left(\hat{\mathbf{b}}(\mathbf{r},\mathbf{e})-\mathbf{b}\right)\left(\hat{\mathbf{b}}(\mathbf{r},\mathbf{e})-\mathbf{b}\right)^H\right] \succeq \mathbf{F}^{-1}(\mathbf{b},\mathbf{e}), \tag{9.7}$$

it is straightforward that

$$E_{\mathbf{e}}E_{\mathbf{r}|\mathbf{b},\mathbf{e}}\left[\left(\hat{\mathbf{b}}(\mathbf{r},\mathbf{e})-\mathbf{b}\right)\left(\hat{\mathbf{b}}(\mathbf{r},\mathbf{e})-\mathbf{b}\right)^H\right] \succeq E_{\mathbf{e}}\left[\mathbf{F}^{-1}(\mathbf{b},\mathbf{e})\right]. \tag{9.8}$$

Noticing that the diagonal submatrices of a semi-positive definite matrix are also semi-positive definite, we have moreover

$$E_{\mathbf{e}}E_{\mathbf{r}|\mathbf{b},\mathbf{e}}\left[\left(\hat{\mathbf{b}}(\mathbf{r},\mathbf{e})-\mathbf{b}\right)^H\left(\hat{\mathbf{b}}(\mathbf{r},\mathbf{e})-\mathbf{b}\right)\right] \geq \sum_i\Big(E_{\mathbf{e}}\left[\mathbf{F}^{-1}(\mathbf{b},\mathbf{e})\right]\Big)_{ii}. \tag{9.9}$$

The right-hand side of (9.9) is obviously a lower bound on the estimation MSE and will be referred in the sequel to as *soft Cramer-Rao bound (SCRB)*, i.e.

$$\mathrm{SCRB}(\mathbf{b}) \triangleq \sum_i\Big(E_{\mathbf{e}}\left[\mathbf{F}^{-1}(\mathbf{b},\mathbf{e})\right]\Big)_{ii}. \tag{9.10}$$

Note that the evaluation of the soft Cramer-Rao bound is quite tedious since it requires to compute the expectation with respect to $\mathbf{e}$ of the inverse of the Fisher information matrix.

It is possible to derive another lower bound, much easier to evaluate in practice, by using the Jensen's inequality for matrices [43]. Indeed, using the latter inequality we have

$$E_{\mathbf{e}}\left[\mathbf{F}^{-1}(\mathbf{b},\mathbf{e})\right] \succeq \Big(E_{\mathbf{e}}\left[\mathbf{F}(\mathbf{b},\mathbf{e})\right]\Big)^{-1}, \tag{9.11}$$

and therefore by the same arguments as above, we have

$$E_{\mathbf{e}}E_{\mathbf{r}|\mathbf{b},\mathbf{e}}\left[\left(\hat{\mathbf{b}}(\mathbf{r},\mathbf{e})-\mathbf{b}\right)^H\left(\hat{\mathbf{b}}(\mathbf{r},\mathbf{e})-\mathbf{b}\right)\right] \geq \sum_i\Big(E_{\mathbf{e}}\left[\mathbf{F}(\mathbf{b},\mathbf{e})\right]\Big)^{-1}_{ii}. \tag{9.12}$$

The right-hand side of (9.12) is a lower bound on the estimation MSE and will be referred in the sequel to as *soft modified Cramer-Rao bound (SMCRB)*, i.e.

$$\mathrm{SMCRB}(\mathbf{b}) \triangleq \sum_i\Big(E_{\mathbf{e}}\left[\mathbf{F}(\mathbf{b},\mathbf{e})\right]\Big)^{-1}_{ii}. \tag{9.13}$$

The SMCRB is usually much easier to evaluate than the SCRB since it does not require to take the expectation of the inverse of a matrix.

Let us now focus on the properties of the SCRB and the SMCRB. We have:

[1] Remind that $\mathbf{A} \succeq \mathbf{B}$ means that $\mathbf{A}-\mathbf{B}$ is semi-positive definite.

1. $\mathrm{SCRB}(\mathbf{b}) \geq \mathrm{SMCRB}(\mathbf{b}) \geq \mathrm{MCRB}(\mathbf{b})$
2. $\lim_{\mathrm{SNR}\to\infty} \mathrm{SCRB}(\mathbf{b}) = \mathrm{MCRB}(\mathbf{b})$
3. $\lim_{\mathrm{SNR}\to\infty} \mathrm{SMCRB}(\mathbf{b}) = \mathrm{MCRB}(\mathbf{b})$
4. $\lim_{K\to\infty} \mathrm{SCRB}(\mathbf{b}) = \mathrm{SMCRB}(\mathbf{b})$ (as long as messages e_k are independent, identically distributed (i.i.d.) variables.)

The first inequality in property 1 is a direct consequence of (9.11). The second inequality is proved in the appendix. Property 2 derives from the high-SNR property of CRBs associated to synchronization parameter estimation: it is shown in [44] that the high SNR asymptote of any CRB is the MCRB. Since this conclusion is valid for any $\mathrm{CRB}(\mathbf{b}, \mathbf{e})$, it is also valid for the $\mathrm{SCRB}(\mathbf{b})$. Property 3 is straightforward from properties 1 and 2. Finally, property 4 is proved in the appendix. This property states that SCRB = SMCRB as long as the frame is long enough and that each message e_k brings an independent information about the corresponding symbol.

Let us finally consider two particular cases of the SCRB and the SMCRB. First assume that the soft information vector brings no information about the the transmitted sequence. In this case, we have

$$p(\mathbf{a}|\mathbf{e}) = \frac{1}{|\mathcal{A}^K|}. \tag{9.14}$$

If (9.14) is true for any soft information vector $\mathbf{e}$, we have

$$\mathrm{SCRB}(\mathbf{b}) = \mathrm{SMCRB}(\mathbf{b}) = \mathrm{CRB}_{\mathrm{NDA}}(\mathbf{b}), \tag{9.15}$$

i.e. both the SCRB and the SMCRB are equal to the CRB computed under a NDA assumption.

Secondly, assume that the vector of soft information gives a perfect knowledge of the transmitted sequence. In this case we have

$$p(\mathbf{r}|\mathbf{b}, \mathbf{e}) = p(\mathbf{r}|\mathbf{b}, \mathbf{a}). \tag{9.16}$$

If (9.16) is verified for any soft-information vector $\mathbf{e}$, we easily have by plugging (9.16) into the expression of the SMCRB, that

$$\mathrm{SMCRB}(\mathbf{b}) = \mathrm{MCRB}(\mathbf{b}). \tag{9.17}$$

The MCRB may therefore be viewed as a SMCRB with a particular distribution of messages $\mathbf{e}$.

9.1.3 Practical Evaluation of the SCRB and the SMCRB

In this section we briefly discuss the practical evaluation of the SCRB and the SMCRB. As expected, the SMCRB is much easier to evaluate than the SCRB.

The Fisher information matrix $\mathbf{F}(\mathbf{r}, \mathbf{e})$ may be computed as follows. Using (9.6) and taking the model of the observations (2.8) into account, we have that the elements of $\nabla \log p(\mathbf{r}|\mathbf{b}, \mathbf{e})$ may be computed according to (7.43)-(7.46). The evaluation of these expressions requires the knowledge of marginal a posteriori probabilities $p(a_k|\mathbf{r}, \mathbf{b}, \mathbf{e})$. Since we assume that any message e_k only contains information about the corresponding symbol a_k, we have

$$p(a_k|\mathbf{r}, \mathbf{b}, \mathbf{e}) \propto p(y_k|a_k, \mathbf{b})\, p(a_k|e_k). \tag{9.18}$$

Posterior probabilities $p(a_k|\mathbf{r}, \mathbf{b}, \mathbf{e})$ are therefore easy to evaluate since only requiring the multiplication of two known probabilities.

The expectation with respect to $\mathbf{e}$ in (9.10) and (9.13) does not lead to any simple analytical expression of the SCRB and the SMCRB. We propose therefore to approximate mathematical expectation $E_{\mathbf{e}}[\cdot]$ by an arithmetical average. In the case of the SMCRB, the arithmetical average performed to approximate the expectation with respect to $\mathbf{e}$ merges the one performed to approximate the expectation with respect to $\mathbf{r}|\mathbf{b}, \mathbf{e}$. As a consequence, overall expectation $E_{\mathbf{e}} E_{\mathbf{r}|\mathbf{b},\mathbf{e}}[\cdot]$ may simply be performed by averaging the Fisher information matrix $\mathbf{F}(\mathbf{r}, \mathbf{e})$ for a large number of independently-generated vectors $\mathbf{r}$ and $\mathbf{e}$. Unlike the SMCRB, the SCRB does not allow to combine the expectations with respect to $\mathbf{e}$ and $\mathbf{r}|\mathbf{b}, \mathbf{e}$ in this way. It is therefore more complicated to compute. We proceed as follows: we draw a realization of $\mathbf{e}$ according to distribution $p(\mathbf{e})$. We then compute the Fisher information matrix $\mathbf{F}(\mathbf{b}, \mathbf{e})$ associated to this particular instance of $\mathbf{e}$ by performing the arithmetical average of factors $\nabla \log p(\mathbf{r}|\mathbf{b}, \mathbf{e})\, \nabla^H \log p(\mathbf{r}|\mathbf{b}, \mathbf{e})$ with respect to $\mathbf{r}|\mathbf{b}, \mathbf{e}$. Repeating this operation for a large number of independently-generated vectors $\mathbf{e}$, we may approximate expectation $E_{\mathbf{e}}[\mathbf{F}^{-1}(\mathbf{b}, \mathbf{e})]$ by an arithmetical average over matrices $\mathbf{F}^{-1}(\mathbf{b}, \mathbf{e})$.

9.2 Lower bound on the speed of convergence

In this section, we derive a lower bound on the speed of convergence of the EM algorithm when it is applied to the estimation of a scalar parameter b.

In the particular case of synchronization, this lower bound is shown to be a function of both the CRB and the MCRB associated the estimation problem.

Dempster, Laird and Rubin showed in their seminal paper [48] that the convergence of the EM algorithm is linear i.e. we have in a neighborhood of $\hat{b}_{\mathrm{ML}}$ that

$$\Delta^{(n+1)} = C(\mathbf{r})\,\Delta^{(n)}, \tag{9.19}$$

where $\Delta^{(n)} = \| \hat{b}^{(n)} - \hat{b}_{ML} \|$ and $C(\mathbf{r})$ is the *rate of convergence* of the EM algorithm. The authors moreover showed that the rate of convergence of the EM algorithm may be related to the amount of *missing information* in the considered problem (see Chapter 6), i.e.

$$C(\mathbf{r}) = I_c^{-1}(\mathbf{r})\, I_m(\mathbf{r}), \tag{9.20}$$

where $I_c(\mathbf{r})$ and $I_m(\mathbf{r})$ are respectively the information matrix of the complete data and the information matrix of the missing data i.e.

$$I_c(\mathbf{r}) \triangleq -\left(\int_{\mathbf{z}} p(\mathbf{z}|\mathbf{r},b)\, \frac{\partial^2}{\partial b^2}\, \log p(\mathbf{z}|b)\, d\mathbf{z} \right)_{\Big| b=\hat{b}_{ML}} \tag{9.21}$$

$$I_m(\mathbf{r}) \triangleq -\left(\int_{\mathbf{z}} p(\mathbf{z}|\mathbf{r},b)\, \frac{\partial^2}{\partial b^2}\, \log p(\mathbf{z}|\mathbf{r},b)\, d\mathbf{z} \right)_{\Big| b=\hat{b}_{ML}}. \tag{9.22}$$

Although explicit, expressions (9.19)-(9.22) do not enable a good intuition of the EM-algorithm behavior in practical problems. In order to derive a more intuitive expression of the speed of convergence of the EM algorithm, let us consider the mean logarithm of $\Delta^{(n)}$ i.e. $E_{\mathbf{r}|b^\star}[\log \Delta^{(n)}]$, where $b^\star$ stands for the actual parameter value. By using (9.19) we have

$$\begin{aligned} E_{\mathbf{r}|b^\star}\left[\log \Delta^{(n)}\right] &= E_{\mathbf{r}|b^\star}\left[\log C(\mathbf{r})\right] + E_{\mathbf{r}|b^\star}\left[\log \Delta^{(n-1)}\right], \\ &= n\, E_{\mathbf{r}|b^\star}\left[\log C(\mathbf{r})\right] + E_{\mathbf{r}|b^\star}\left[\log \Delta^{(0)}\right], \end{aligned} \tag{9.23}$$

where $\Delta^{(0)} = \| \hat{b}^{(0)} - \hat{b}_{ML} \|$. From (9.23), it appears clearly that factor $E_{\mathbf{r}|b^\star}\left[\log C(\mathbf{r})\right]$ is a measure of the (mean) speed of convergence of the EM estimates $\hat{b}^{(n)}$ to the ML estimate $\hat{b}_{ML}$. In the appendix we show that

$$E_{\mathbf{r}|b^\star}\left[\log C(\mathbf{r})\right] \geq \log\Big(1 - \mathrm{CRB}_{\mathbf{z}}(b^\star)\, \mathrm{CRB}_{\mathbf{r}}^{-1}(b^\star)\Big), \tag{9.24}$$

as long as $\mathrm{CRB}_{\mathbf{r}}(b_{ML})$ is sufficiently small. In this expression, $\mathrm{CRB}_{\mathbf{z}}$ represents the Cramer-Rao bound relative to the estimation of parameter b when vector

$\mathbf{z}$ is at our disposal. Similarly, $\text{CRB}_{\mathbf{r}}$ is the bound on the achievable estimation accuracy given the actual observation vector $\mathbf{r}$.

Combining (9.24) with (9.23), we get

$$E_{\mathbf{r}|b^\star}\left[\log \Delta^{(n)}\right] \geq n \log\left(1 - \text{CRB}_{\mathbf{z}}(b^\star)\,\text{CRB}_{\mathbf{r}}^{-1}(b^\star)\right) + E_{\mathbf{r}|b^\star}\left[\log \Delta^{(0)}\right]. \quad (9.25)$$

From (9.25) we may therefore notice that the distance between the EM estimates $\hat{b}^{(n)}$ and the convergence point $\hat{b}_{ML}$ decreases as a function of the ratio $\text{CRB}_{\mathbf{z}}/\text{CRB}_{\mathbf{r}}$. This ratio actually represents the improvement of the variance estimation error which can be achieved by observing the complete-data set instead of the incomplete-data set. Expression (9.25) therefore relates results about estimation accuracy to the speed of convergence of the EM algorithm. In particular we notice that the more information is contained in the missing data, i.e. the more $\text{CRB}_{\mathbf{z}} < \text{CRB}_{\mathbf{r}}$, the slower the EM algorithm will converge.

Let us now particularize (9.25) to the convergence of the EM-based synchronizer derived in Section 7.4.1. In that case, the complete data set is defined as $\mathbf{z} \triangleq [\mathbf{r}^T, \mathbf{a}^T]^T$ and it follows that

$$\text{CRB}_{\mathbf{z}}(b) = \text{MCRB}(b). \quad (9.26)$$

Indeed, by definition of the Cramer-Rao bound, we may write

$$\text{CRB}_{\mathbf{z}}(b) = -\int_{\mathbf{z}} p(\mathbf{z}|b)\,\frac{\partial^2}{\partial b^2}\log p(\mathbf{z}|b)\,d\mathbf{z} \quad (9.27)$$

Using the definition of the complete data set $\mathbf{z}$, we have

$$\begin{aligned}\text{CRB}_{\mathbf{z}}(b) &= -\sum_{\mathbf{a}}\int_{\mathbf{r}} p(\mathbf{r},\mathbf{a}|b)\,\frac{\partial^2}{\partial b^2}\log p(\mathbf{r},\mathbf{a}|b)\,d\mathbf{r},\\ &= -\sum_{\mathbf{a}}\int_{\mathbf{r}} p(\mathbf{r},\mathbf{a}|b)\,\frac{\partial^2}{\partial b^2}\Big(\log p(\mathbf{r}|\mathbf{a},b) + \log p(\mathbf{a})\Big)\,d\mathbf{r}, \end{aligned} \quad (9.28)$$

where the last equality follows from both the Bayes rule and the independence between $\mathbf{a}$ and b. Dropping the term independent of b in the derivation factor leads to

$$\text{CRB}_{\mathbf{z}}(b) = -\sum_{\mathbf{a}}\int_{\mathbf{r}} p(\mathbf{r},\mathbf{a}|b)\,\frac{\partial^2}{\partial b^2}\Big(\log p(\mathbf{r}|\mathbf{a},b)\Big)\,d\mathbf{r}. \quad (9.29)$$

Comparing (9.29) with the definition of the modified CRB (3.51), we have (9.26).

Plugging (9.26) into (9.25), we then get

$$E_{\mathbf{r}|b^\star}\left[\log \Delta^{(n)}\right] \geq n \log\left(1 - \mathrm{MCRB}(b^\star)\,\mathrm{CRB}_{\mathbf{r}}^{-1}(b^\star)\right) + E_{\mathbf{r}|b^\star}\left[\log \Delta^{(0)}\right]. \quad (9.30)$$

The convergence of the EM algorithm is therefore related to both the Cramer-Rao bound and modified Cramer-Rao bound associated to estimation problem (7.2). Note that, using the fact that $\mathrm{MCRB}(b) \leq \mathrm{CRB}_{\mathbf{r}}(b)$ [42], we have

$$0 < \mathrm{MCRB}(b)\,\mathrm{CRB}_{\mathbf{r}}^{-1}(b) \leq 1, \quad (9.31)$$

and therefore,

$$\log\left(1 - \mathrm{MCRB}(b)\,\mathrm{CRB}_{\mathbf{r}}^{-1}(b)\right) < 0. \quad (9.32)$$

This result is in accordance with the convergence behavior of the EM algorithm since (9.32) implies that, at each iteration, each new EM estimate gets closer to the ML estimate. The speed at which these estimates converge to $\hat{b}_{ML}$ depends however on the ratio $\mathrm{MCRB}(b)\,\mathrm{CRB}_{\mathbf{r}}^{-1}(b)$. On the one hand, $\mathrm{MCRB}(b)\,\mathrm{CRB}_{\mathbf{r}}^{-1}(b) = 1$ means that the EM algorithm is able to compute the ML estimate in only one step. Note that $\mathrm{CRB}_{\mathbf{r}} \simeq \mathrm{MCRB}$ may occur either when we have a perfect a priori knowledge about the transmitted sequence or when the system operates at sufficiently high SNR [44]. On the other hand, $\mathrm{MCRB}(b)\,\mathrm{CRB}_{\mathbf{r}}^{-1}(b) \simeq 0$ will lead to a very slow convergence of the EM algorithm. This kind of situations occurs when there is a huge gap between the estimation quality which can be achieved with and without the knowledge of the transmitted sequence.

9.3 Main Points

In this chapter we derive lower bounds on the synchronizer performance:

- We first attempt to bound the MSE achievable by any soft-information-directed synchronizer. We derive the so-called soft Cramer-Rao bound (SCRB) and soft modified Cramer-Rao bound (SMCRB). These bounds have the important properties described in section 9.1.2. In particular, the SMCRB is usually looser than the SCRB but is also much easier to evaluate. At high SNR, both the SCRB and the SMCRB tend to the modified Cramer-Rao bound (MCRB). When the frame length tends to infinity and provided that the soft-information messages are i.i.d., the SMCRB tends to the SCRB.

- We then derive a lower bound on the speed of convergence of the EM synchronizer (see section 9.2). This lower bound is both a function of the Cramer-Rao bound (CRB) and the modified CRB (MCRB) associated to the estimation problem. It enables a good intuition of the EM-algorithm behavior in practical problems. In particular, we emphasize that the EM synchronizer may exhibit a low speed of convergence when there is a significant gap between the CRB and the MCRB.

9.4 Appendices

9.4.1 Proof that $\mathrm{SMCRB}(\mathbf{b}) \geq \mathrm{MCRB}(\mathbf{b})$

In this section we prove that

$$\mathrm{SMCRB}(\mathbf{b}) \geq \mathrm{MCRB}(\mathbf{b}). \tag{9.33}$$

To prove our claim we will show that

$$E_{\mathbf{e}}\Big[\mathbf{F}(\mathbf{b},\mathbf{e})\Big] \preceq E_{\mathbf{a}}\Big[\mathbf{G}(\mathbf{b},\mathbf{a})\Big], \tag{9.34}$$

where $\mathbf{G}(\mathbf{b},\mathbf{a})$ has been defined in (3.51).

First, we have by using the definition of the Fisher information matrix that

$$\begin{aligned} E_{\mathbf{e}}\Big[\mathbf{F}(\mathbf{b},\mathbf{e})\Big] &= E_{\mathbf{e}}\, E_{\mathbf{r}|\mathbf{b},\mathbf{e}}\Big[\nabla \log p(\mathbf{r}|\mathbf{b},\mathbf{e})\, \nabla^H \log p(\mathbf{r}|\mathbf{b},\mathbf{e})\Big], \\ &= E_{\mathbf{e}}\left[\int \Big(\nabla \log p(\mathbf{r}|\mathbf{b},\mathbf{e})\, \nabla^H \log p(\mathbf{r}|\mathbf{b},\mathbf{e})\Big)\, p(\mathbf{r}|\mathbf{b},\mathbf{e})\, d\mathbf{r}\right] \end{aligned} \tag{9.35}$$

Taking into account that

$$\nabla \log p(\mathbf{r}|\mathbf{b},\mathbf{e}) = \frac{1}{p(\mathbf{r}|\mathbf{b},\mathbf{e})}\, \nabla p(\mathbf{r}|\mathbf{b},\mathbf{e}), \tag{9.36}$$

we have also that

$$E_{\mathbf{e}}\Big[\mathbf{F}(\mathbf{b},\mathbf{e})\Big] = E_{\mathbf{e}}\left[\int \Big(\nabla p(\mathbf{r}|\mathbf{b},\mathbf{e})\, \nabla^H p(\mathbf{r}|\mathbf{b},\mathbf{e})\Big)\, \frac{1}{p(\mathbf{r}|\mathbf{b},\mathbf{e})}\, d\mathbf{r}\right]. \tag{9.37}$$

Let us now expand probability $p(\mathbf{r}|\mathbf{b},\mathbf{e})$ as (9.3) i.e.

$$p(\mathbf{r}|\mathbf{b},\mathbf{e}) = \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{a}|\mathbf{e}). \tag{9.38}$$

Since $p(\mathbf{a}|\mathbf{e})$ does not depend on $\mathbf{b}$, we get by applying the gradient operator to (9.38):

$$\begin{aligned}\nabla p(\mathbf{r}|\mathbf{b},\mathbf{e}) &= \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{e})\,\nabla p(\mathbf{r}|\mathbf{a},\mathbf{b}) \\ &= \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{e})\,p(\mathbf{r}|\mathbf{a},\mathbf{b})\,\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \qquad (9.39)\end{aligned}$$

where (9.39) have been obtained by considering the same kind of equality as (9.36). Let us rewrite (9.39) as

$$\nabla p(\mathbf{r}|\mathbf{b},\mathbf{e}) = \sum_{\mathbf{a}} \sqrt{p(\mathbf{a}|\mathbf{e})\,p(\mathbf{r}|\mathbf{a},\mathbf{b})}\,\sqrt{p(\mathbf{a}|\mathbf{e})\,p(\mathbf{r}|\mathbf{a},\mathbf{b})}\,\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b}). \quad (9.40)$$

Multiplying both sides of (9.40) by an arbitrary vector v^H, we get the following scalar expression:

$$v^H\,\nabla p(\mathbf{r}|\mathbf{b},\mathbf{e}) = \sum_{\mathbf{a}} \sqrt{p(\mathbf{a}|\mathbf{e})\,p(\mathbf{r}|\mathbf{a},\mathbf{b})}\,\sqrt{p(\mathbf{a}|\mathbf{e})\,p(\mathbf{r}|\mathbf{a},\mathbf{b})}\,v^H\,\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b}). \quad (9.41)$$

Squaring both sides of (9.41) and applying the Schwarz inequality, we get

$$\begin{aligned}v^H\nabla p(\mathbf{r}|\mathbf{b},\mathbf{e})\nabla^H p(\mathbf{r}|\mathbf{b},\mathbf{e})\,v \le &\sum_{\mathbf{a}} v^H\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\nabla^H \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\,v \\ &\times p(\mathbf{r}|\mathbf{a},\mathbf{b})\,p(\mathbf{a}|\mathbf{e}) \\ &\times \sum_{\mathbf{a}} p(\mathbf{a}|\mathbf{e})\,p(\mathbf{r}|\mathbf{b},\mathbf{e}). \\ = &\sum_{\mathbf{a}} v^H\nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\nabla^H \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\,v \\ &\times p(\mathbf{r}|\mathbf{a},\mathbf{b})\,p(\mathbf{a}|\mathbf{e})\,p(\mathbf{r}|\mathbf{b},\mathbf{e}). \qquad (9.42)\end{aligned}$$

Since inequality (9.42) is true for any arbitrary vector v, we have

$$\begin{aligned}\nabla p(\mathbf{r}|\mathbf{b},\mathbf{e})\nabla^H p(\mathbf{r}|\mathbf{b},\mathbf{e}) \preceq &\sum_{\mathbf{a}} \nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\nabla^H \log p(\mathbf{r}|\mathbf{a},\mathbf{b}) \\ &\times p(\mathbf{r}|\mathbf{a},\mathbf{b})\,p(\mathbf{a}|\mathbf{e})\,p(\mathbf{r}|\mathbf{b},\mathbf{e}). \qquad (9.43)\end{aligned}$$

Substituting this result into (9.37), we have

$$\begin{aligned}E_{\mathbf{e}}\Big[\mathbf{F}(\mathbf{b},\mathbf{e})\Big] &\preceq \sum_{\mathbf{a}} \int \nabla \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\nabla^H \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\,p(\mathbf{r}|\mathbf{b},\mathbf{a})\,p(\mathbf{a})\,d\mathbf{r}, \\ &= E_{\mathbf{a}} E_{\mathbf{r}|\mathbf{b},\mathbf{a}}\Big[\log p(\mathbf{r}|\mathbf{a},\mathbf{b})\nabla^H \log p(\mathbf{r}|\mathbf{a},\mathbf{b})\Big], \\ &= E_{\mathbf{a}}\Big[\mathbf{G}(\mathbf{b},\mathbf{a})\Big],\end{aligned}$$

which proves (9.34).

9.4.2 Proof that $\lim_{K\to\infty} \mathrm{SCRB}(\mathbf{b}) = \mathrm{SMCRB}(\mathbf{b})$

We want to show that

$$\lim_{K\to\infty} \mathrm{SMCRB}(\mathbf{b}) = \mathrm{SCRB}(\mathbf{b}), \tag{9.44}$$

as long as the messages e_k are independent, identically distributed (i.i.d.) variables. The demonstration is based on the strong law of large numbers [66]. This law states that for i.i.d. variables x_k, $\lim_{K\to\infty} \frac{1}{K}\sum_k^K x_k$ is equal with probability 1 to its expected value $E[x]$. Using this result, we first show that the factors $\nabla \log p(\mathbf{r}|\mathbf{b},\mathbf{e})$ making up the Fisher information matrix are equal to a constant with probability 1.

Let us consider one particular element of $\nabla \log p(\mathbf{r}|\mathbf{b})$, say $\frac{\partial \log p(\mathbf{r}|\mathbf{b},\mathbf{e})}{\partial\theta}$. From (7.45) we have

$$\frac{\partial}{\partial\theta}\log p(\mathbf{r}|\mathbf{b}) = \frac{A}{N_0}\,\mathrm{Im}\Big\{\sum_{k=0}^{K-1}\eta_k^*\, y_k(\nu,\tau)\, e^{-j\theta}\Big\}, \tag{9.45}$$

where η_k is defined as

$$\eta_k = \sum_a a\, p(a_k = a|\mathbf{r},\mathbf{b},\mathbf{e}), \tag{9.46}$$

and

$$p(a_k|\mathbf{r},\mathbf{b},\mathbf{e}) \propto p(y_k|a_k,\mathbf{b})\, p(a_k|e_k). \tag{9.47}$$

A posteriori mean value η_k therefore only depends on current matched-filter output y_k and soft-information message e_k. Taking then into account that matched filter output $y(kT+\tau)$ may be written as[2]

$$y(kT+\tau) = a_k e^{j\theta} + w_k,$$

where w_k is a zero-mean white Gaussian noise, we have

$$\frac{\partial \log p(\mathbf{r}|\mathbf{b},\mathbf{e})}{\partial\theta} = \frac{A}{N_0}\mathrm{Im}\Big\{\sum_k \eta_k^*\, a_k + w_k\, e^{j\theta}\Big\}. \tag{9.48}$$

If soft-information messages e_k are i.i.d., all the terms in (9.48) are i.i.d and we may therefore apply the strong law of large number:

$$\begin{aligned}\lim_{K\to\infty}\left(\frac{1}{K}\frac{\partial \log p(\mathbf{r}|\mathbf{b},\mathbf{e})}{\partial\theta}\right) &\overset{\mathrm{WP1}}{\to} -\frac{A}{N_0}\mathrm{Im}\Big\{E[\eta_k\, a_k^*] + E[w_k^*]\, e^{-j\theta}\Big\},\\ &\overset{\mathrm{WP1}}{\to} -\frac{A}{N_0}\mathrm{Im}\Big\{E[\eta_k\, a_k^*]\Big\}.\end{aligned} \tag{9.49}$$

[2] Since evaluated with the correct synchronization-parameter values

where the notation $\overset{\text{WP1}}{\rightarrow}$ denotes the convergence with probability 1. The previous reasoning may be extended to the other elements[3] of $\nabla \log p(\mathbf{r}|\mathbf{b},\mathbf{e})$. Therefore, taking (9.4) into account, we may also write

$$\lim_{K\to\infty} \frac{1}{K^2}\mathbf{F}(\mathbf{b},\mathbf{e}) \overset{\text{WP1}}{\rightarrow} \mathbf{C} \tag{9.50}$$

where $\mathbf{C}$ is a constant matrix. Denoting by $\mathcal{E}$ the set of vector $\mathbf{e}$ such that $\frac{1}{K^2}\mathbf{F}(\mathbf{e}) = \mathbf{C}$, equation (9.50) may also be expressed as

$$\lim_{K\to\infty} p\big(\mathbf{e} \in \mathcal{E}\big) = 1 \tag{9.51}$$

i.e. for large value of K, the probability to observe a vector $\mathbf{e}$ such that $\frac{1}{K^2}\mathbf{F}(\mathbf{e},\mathbf{b}) = \mathbf{C}$, is certain with probability 1.

Let us now consider the expression of the SMCRB in (9.13) i.e.

$$\text{SMCRB} = \Big(E_{\mathbf{e}}\big[\mathbf{F}(\mathbf{e},\mathbf{b})\big]\Big)^{-1} \tag{9.52}$$

Since the mathematical expectation is a linear operation, we may also write

$$\text{SMCRB} = \Big(E_{\mathbf{e}\in\mathcal{E}}\big[\mathbf{F}(\mathbf{e},\mathbf{b})\big] + E_{\mathbf{e}\notin\mathcal{E}}\big[\mathbf{F}(\mathbf{e},\mathbf{b})\big]\Big)^{-1}. \tag{9.53}$$

Now, as long as K is large enough, we have from (9.51) that $p(\mathbf{e}) \simeq 0 \quad \forall\, \mathbf{e} \notin \mathcal{E}$. Therefore, (9.53) becomes

$$\begin{aligned}\text{SMCRB} &= \Big(E_{\mathbf{e}\in\mathcal{E}}\big[\mathbf{F}(\mathbf{e},\mathbf{b})\big]\Big)^{-1}, \\ &= K^2\,\mathbf{C}^{-1}\end{aligned} \tag{9.54}$$

where the last equality comes from the definition of $\mathcal{E}$. Following the same reasoning, we may show that

$$\text{SCRB} = K^2\,\mathbf{C}^{-1} \qquad \text{for } K \to \infty. \tag{9.55}$$

Comparing (9.54) and (9.55), we finally prove (9.44).

9.4.3 Proof of (9.24)

In this appendix, we show that

$$E_{\mathbf{r}|b^\star}\big[\log C(\mathbf{r})\big] \geq \log\big(1 - \text{CRB}_{\mathbf{z}}(b^\star)\,\text{CBR}_{\mathbf{r}}^{-1}(b^\star)\big) \tag{9.56}$$

[3] The cases of τ and ν are slightly more tricky

as long as $\mathrm{CBR}_{\mathbf{r}}(\hat{b}_{ML})$ is small enough. Using the definition (9.20) of $C(\mathbf{r})$ we may first write

$$E_{\mathbf{r}|b^\star}\left[\log C(\mathbf{r})\right] = E_{\mathbf{r}|b^\star}\left[\log I_c^{-1}(\mathbf{r})\right] + E_{\mathbf{r}|b^\star}\left[\log I_m(\mathbf{r})\right] \tag{9.57}$$

Using then the Jensen's inequality we have

$$\begin{aligned} E_{\mathbf{r}|b^\star}\left[\log C(\mathbf{r})\right] &\geq \log E_{\mathbf{r}|b^\star}\left[I_c^{-1}(\mathbf{r})\right] + \log E_{\mathbf{r}|b^\star}\left[I_m(\mathbf{r})\right] \\ &\geq \log\Big(E_{\mathbf{r}|b^\star}\left[I_c(\mathbf{r})\right]\Big)^{-1} + \log E_{\mathbf{r}|b^\star}\left[I_m(\mathbf{r})\right] \end{aligned} \tag{9.58}$$

In order to prove (9.56), we will show that

$$E_{\mathbf{r}|b^\star}\left[I_c(\mathbf{r})\right] \simeq \mathrm{CRB}_{\mathbf{z}}^{-1}(b^\star) \tag{9.59}$$

$$E_{\mathbf{r}|b^\star}\left[I_m(\mathbf{r})\right] \simeq \mathrm{CRB}_{\mathbf{z}}^{-1}(b^\star) - \mathrm{CRB}_{\mathbf{r}}^{-1}(b^\star) \tag{9.60}$$

as long as $\mathrm{CRB}_{\mathbf{r}}(\hat{b}_{ML})$ is small enough. Let us first consider $E_{\mathbf{r}|b^\star}\left[I_c(\mathbf{r})\right]$. Using the definition (9.21) of the complete-data information matrix $I_c(\mathbf{r})$, we have

$$E_{\mathbf{r}|b^\star}\left[I_c(\mathbf{r})\right] = -\int_{\mathbf{r}} p(\mathbf{r}|b^\star) \int_{\mathbf{z}} p(\mathbf{z}|\mathbf{r},b)\ \frac{\partial^2}{\partial b^2}\ \log p(\mathbf{z}|b)\, d\mathbf{z}\, d\mathbf{r}_{|b=\hat{b}_{ML}}$$

If we assume that $\mathrm{CRB}_{\mathbf{r}}$ is small, we have $\hat{b}_{ML} \simeq b^\star$ and therefore

$$\begin{aligned} E_{\mathbf{r}|b^\star}\left[I_c(\mathbf{r})\right] &\simeq -\int_{\mathbf{r}} p(\mathbf{r}|b^\star) \int_{\mathbf{z}} p(\mathbf{z}|\mathbf{r},b)\ \frac{\partial^2}{\partial b^2}\ \log p(\mathbf{z}|b)\, d\mathbf{z}\, d\mathbf{r}_{|b=b^\star} \\ &= -\int_{\mathbf{z}} p(\mathbf{z}|b)\ \frac{\partial^2}{\partial b^2}\ \log p(\mathbf{z}|b)\, d\mathbf{z}_{|b=b^\star} \\ &= \mathrm{CRB}_{\mathbf{z}}^{-1}(b^\star) \end{aligned} \tag{9.61}$$

where (9.61) follows from the definition of the CRB.

Let us now consider $\log E_{\mathbf{r}|b^\star}\left[I_m(\mathbf{r})\right]$. First notice that $I_m(\mathbf{r})$ and $I_c(\mathbf{r})$ may be related as follows [51]:

$$-\left(\frac{\partial}{\partial b^2}\log p(\mathbf{r}|b)\right)_{\Big|b=\hat{b}_{ML}} = I_c(\mathbf{r}) - I_m(\mathbf{r}). \tag{9.62}$$

Using this expression, we may then write

$$E_{\mathbf{r}|b^\star}\left[I_m(\mathbf{r})\right] = E_{\mathbf{r}|b^\star}\left[I_c(\mathbf{r}) + \left(\frac{\partial}{\partial b^2}\log p(\mathbf{r}|b)\right)_{\Big|b=\hat{b}_{ML}}\right]. \tag{9.63}$$

Assuming that $\mathrm{CRB}_{\mathbf{r}}$ is small and therefore $\hat{b}_{ML} \simeq b^\star$, we have

$$E_{\mathbf{r}|b^\star}\left[I_m(\mathbf{r})\right] = \mathrm{CRB}_{\mathbf{z}}^{-1}(b^\star) - E_{\mathbf{r}|b^\star}\left[\left(\frac{\partial}{\partial b^2}\log p(\mathbf{r}|b)\right)_{\Big|b=b^\star}\right], \tag{9.64}$$

$$= \mathrm{CRB}_{\mathbf{z}}^{-1}(b^\star) - \mathrm{CRB}_{\mathbf{r}}^{-1}(b^\star), \tag{9.65}$$

where (9.64) follows from (9.61) and (9.65) follows from the definition of the CRB. Plugging (9.61) and (9.65) into (9.58) we prove (9.56).

Simulation Results

10

In this chapter we provide simulation results which illustrate the performance of the algorithms derived in the previous chapters.

The chapter is organized as follows. The first section is dedicated to the soft lower bounds derived in chapter 9, namely the SCRB and the SMCRB. Two cases are considered. In the first case, we compute the SCRB and SMCRB using a soft-information distribution which verifies the assumptions made in chapter 9. Using this distribution, the main properties of the SCRB and SMCRB (see section 9.1.2) are illustrated by simulation results. In the second case, we consider the behavior of the SMCRB when the soft information is delivered by an actual turbo receiver.

In a second section, we deal with the performance of the soft-decision-directed (SDD) synchronizers derived in chapter 7. In particular, we study their speed of convergence, ultimate accuracy and acquisition range. Regarding the speed of convergence, we emphasize that the bound on the rate of convergence of the EM synchronizer derived in chapter 9 is tight and gives therefore significant insights into the convergence behavior of the EM synchronizer. We then apply the proposed algorithms to the synchronization of different kind of receivers. We consider the synchronization of a convolutionally-coded, a turbo-coded and a bit-interleaved coded-modulation transmission.

In a last section, we study the performance of the discrete-parameter synchronizers proposed in chapter 8. We only provide preliminary results showing the powerfulness of the proposed algorithms. In particular, we illus-

trate the detection ability of CA MAP synchronizer. The performance of the proposed algorithms is then illustrated in the case of a convolutionally-coded transmission.

10.1 Lower Bounds for Soft-information-directed Synchronizers

In chapter 9 we have derived lower bounds on the MSE achievable by synchronizers handling a vector of symbol soft information. In this section we illustrate the behavior of these bounds by simulation results. In a first subsection, we consider that the soft information is generated according to a Gaussian distribution. In a second subsection, we focus on the bound behavior when the soft information vector is delivered by an actual turbo receiver.

10.1.1 Theoretical Soft-information Distribution

We assume that the symbol soft information is delivered under the form of log-likelihood ratios (LLRs) on the coded bits. We have that

$$p(a_k|e_k) = \prod_{l=1}^{\log_2|\mathcal{A}|} p(x_{k,l}|e_{k,l}), \tag{10.1}$$

where $x_{k,l}$ denotes the l^{th} coded bit making up the k^{th} transmitted symbols and $e_{k,l}$ the associated LLR. Moreover, we make the common assumption [83] that the distribution of messages $e_{k,l}$ given coded bit $x_{k,l}$ is Gaussian i.e.

$$p(e_{k,l}|x_{k,l}) \sim \mathcal{N}(\mu\, x_{k,l}\,, \sigma^2) \qquad \text{with } \mu = \sigma^2/2, \tag{10.2}$$

where μ and σ^2 respectively denotes the mean and the variance of the distribution. Since the mean and the variance of the distribution are related, $p(e_{k,l}|x_{k,l})$ may be parameterized by only one parameter. In the sequel we choose to parameterize the LLR distribution by the mutual information $I(e_{k,l}; x_{k,l})$ between $e_{k,l}$ and the corresponding coded bit $x_{k,l}$. This mutual information is defined

as:

$$I(e_{k,l}\,;x_{k,l}) = \frac{1}{2}\sum_{c\in\{+1,-1\}}\int_{-\infty}^{+\infty} p(e_{k,l}=e|x_{k,l}=c) \times \log_2 \frac{2\,p(e_{k,l}=e|x_{k,l}=c)}{p(e_{k,l}=e|x_{k,l}=+1)+p(e_{k,l}=e|x_{k,l}=-1)}de,$$

where we have assumed that $p(x_{k,l}) \simeq \frac{1}{2}$. There is a univoque relation between $I(e_{k,l}\,;x_{k,l})$ and σ^2 [83]. Mutual information ranges from 0 to 1. The higher the mutual information, the more information about the transmitted symbols. For example, if $I(e_{k,l}\,;x_{k,l}) = 0$ it is easy to show that $\mu = 0$ and $\sigma^2 = 0$. All the messages (i.e. the LLRs) are therefore equal to 0 and we have no information about the transmitted symbols. On the contrary, when $I(e_{k,l}\,;x_{k,l}) = 1$, the LLRs provide a perfect knowledge of the transmitted symbols.

The messages generated according to (10.2) verifies the hypotheses made in chapter 9 for the derivation of the SCRB and the SMCRB. In this subsection we will illustrate by simulation results the properties of the SCRB and SMCRB (see section 9.1.2) by generating the soft symbol messages according to (10.2). We consider the case of carrier phase offset estimation.

Illustration of properties 1 and 4

Fig. 10.1 represents the ratio SCRB/SMCRB versus the frame length for different values of mutual information $I(e_{k,l};x_{k,l})$. The curves are plotted for the case of carrier phase offset estimation and considering QAM-64 transmission with Gray mapping. The E_s/N_0-ratio is set to 0 dB. We consider mutual information $I(e_{k,l};x_{k,l})$ equal to 0, 0.5 and 1. We may observe that all the curves are higher or equal to 0 dB. This means, as mentioned by property 1 of section 9.1.2, that the SCRB is always greater or equal to the SMCRB. Note also that the ratio SCRB/SMCRB is always equal to 0 dB for $I(e_{k,l};x_{k,l}) = 0$. Indeed, in this case, the soft messages contain no information about the transmitted symbols and therefore, as mentioned in section 9.1.2, SCRB = SMCRB = $\mathrm{CRB}_{\mathrm{NDA}}$. In the general case, the SMCRB is looser than the SCRB for short frame lengths. The asymptotical properties of the SMCRB mentioned by property 4 of section 9.1.2 appears however quite rapidly as the number of transmitted symbols increases. In the case we consider for example, the equality between the SMCRB

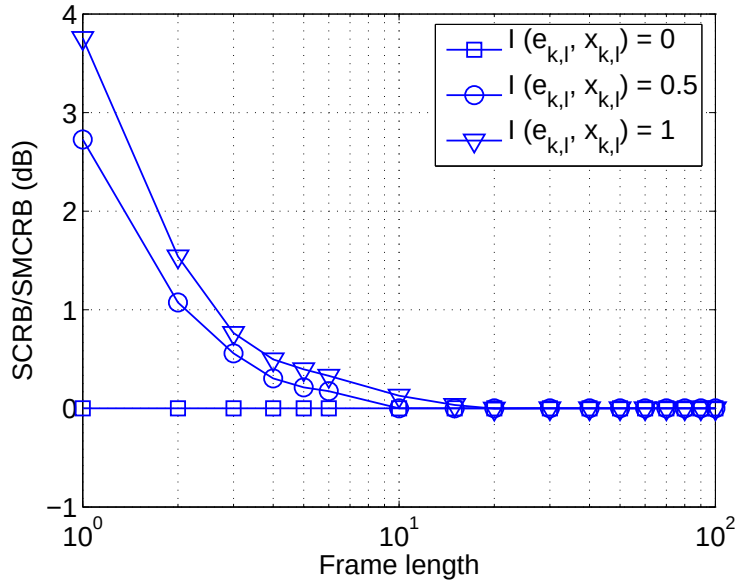

Figure 10.1 SCRB/SMCRB-ratio (in dB) associated to carrier phase offset estimation versus the frame length. We consider a 64-QAM modulation with gray mapping and $E_s/N_0 = 0$ dB. Curves are plotted for different quantities of a priori information.

and the SCRB is already reached for a frame length larger than 15 symbols. In practice, the number of transmitted symbols is typically higher than 100. Therefore, the SMCRB, which is easier to compute than the SCRB, may be used in practice as an equivalent benchmark for assessing the performance of soft estimators.

Illustration of properties 1 and 3

Fig. 10.2 represents the SMCRB associated to the carrier phase estimation versus the E_s/N_0-ratio. Again, for the sake of conciseness, only the results relative to the carrier phase offset estimation are presented. The frame length is set to 500 QPSK symbols. Curves are plotted for mutual information equal to 0, 0.2, 0.8 and 1.

Remember (see section 9.1.2) that the case $I(e_{k,l}; x_{k,l}) = 0$ actually corresponds to the NDA CRB. The curve relative to $I(e_{k,l}; x_{k,l}) = 1$ corresponds to the MCRB. Indeed, in the latter case, soft symbol messages provide a perfect knowledge of the transmitted symbols. The intermediate cases show how far

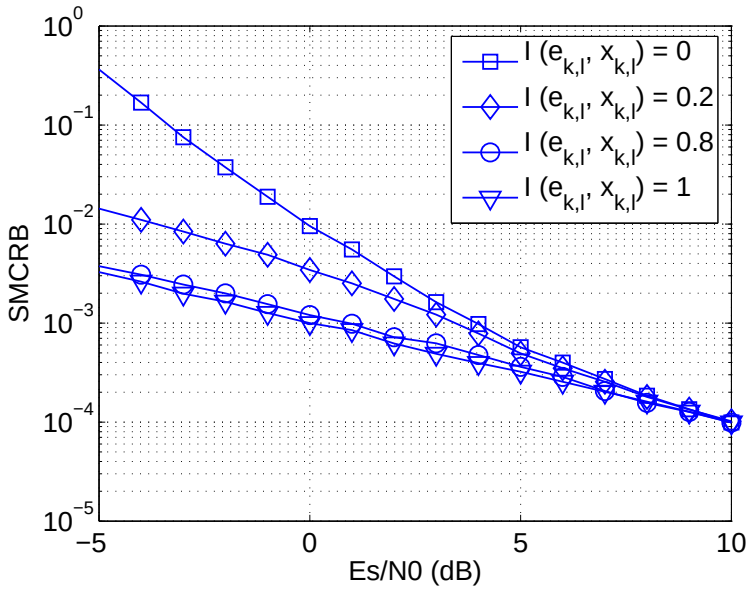

Figure 10.2 SMCRB associated to carrier phase offset estimation versus E_s/N_0-ratio. The frames consit in 500 QPSK symbols. The curves are plotted for different amounts of a priori symbol information.

we are from these two limit cases when we have some information about the transmitted symbols at our disposal. From Fig. 10.2 it is clear that the SMCRB depends on the quantity of information about the symbols which is available: the more information, the lower the SMCRB. Note that we have always that SMCRB $\geq$ MCRB, which is in accordance with property 1 of section 9.1.2.

At high SNR, we may also notice that property 3 is verified: all the SMCRBs tend to be equal to the MCRB as long as the SNR is high enough. In other words, there is no interest in having some symbol soft information at our disposal in this range of SNRs. At lower SNR however, we see that we can benefit from the presence of some information about the transmitted symbols. State-of-the-art receivers, such as the turbo-receivers for example, operate in this range of SNRs. Taking into account the available soft information in the synchronization process seems therefore to be relevant in this kind of receivers.

10.1.2 Soft-information Delivered by a Turbo Receiver

In this subsection we focus on the behavior of the SMCRB when the soft information vector $\mathbf{e}$ is delivered by a turbo receiver. In this case, the SCRB may no longer be easily computed and will not be considered hereafter. We assume that the soft information vector handled by the synchronizer is made up with the coded-bit extrinsic LLRs delivered at a given iteration by the turbo receiver. Note that the assumption of independence between the observations $\mathbf{r}$ and the soft-information vector $\mathbf{e}$ does no longer hold here since the extrinsic LLRs are computed from the observation vector $\mathbf{r}$. As a consequence, the SMCRB may no longer be proved to be an accurate lower bound. However, as far as our simulation setups are concerned, we have always observed that the SMCRB is tight as long as the frame length is large enough. The SMCRB may therefore be used to have some insights into the achievable performance of SDD synchronizers.

When considering soft lower bounds computed from the extrinsic information delivered by a turbo receiver, a crucial point is the *statistical consistency* of this soft information. By statistical consistency, we mean that

$$\lim_{l\to\infty} \frac{N_l(a|e)}{\sum_{a\in\mathcal{A}} N_l(a|e)} = p(a|e) \tag{10.3}$$

where $N_l(a|e)$ denotes the number (over l experiments) of transmitted symbols equal to a when soft message e has been observed. In other words, the statistical consistency ensures that the actual symbol distribution corresponds to the distribution predicted from the soft message e. Any information which would not be statistically consistent would obviously lead to inaccurate lower bounds since providing bad statistical information about the transmitted symbols. As long as our simulation scenari are concerned, we have therefore verified by Monte-Carlo simulations that the soft information delivered by a perfectly synchronized turbo receiver is statistically consistent. On the contrary, the soft information delivered by an unsynchronized turbo receiver is obviously not. In the sequel we will therefore consider the SMCRBs computed from the soft information delivered by a perfectly-synchronized system.

We focus on the case of carrier-phase synchronization for an unpunctured turbo-coded transmission. The turbo encoder consists of two rate-1/2 recursive systematic convolutional (RSC) encoders with polynomial generators $(21, 37)_8$. The coded bits are mapped onto a QPSK signaling constellation

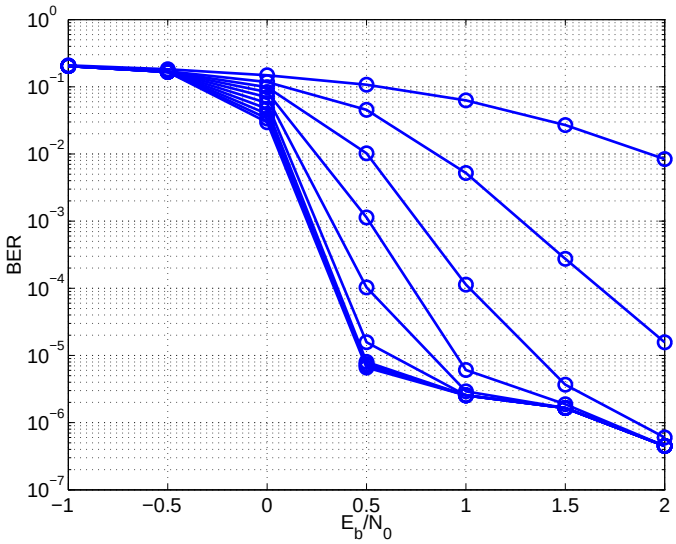

Figure 10.3 BER versus the E_b/N_0-ratio for a turbo-coded transmission. The turbo encoder consists of two rate-1/2 RSC encoders with polynomial generators $(21, 37)_8$. The transmitted frames are made up with 9999 Gray-mapped QPSK symbols.

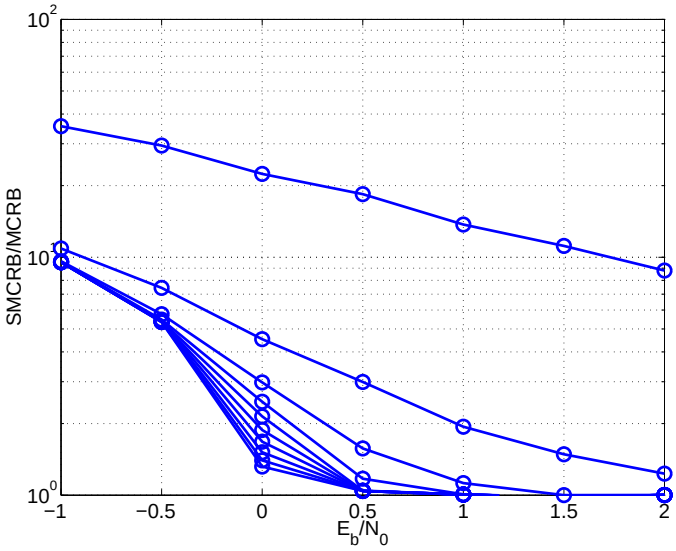

Figure 10.4 SMCRB/MCRB (associated to carrier phase estimation) versus the E_b/N_0-ratio for a turbo-coded transmission. The SMCRB is computed from the extrinsic probabilities delivered by the turbo decoder.

with Gray mapping. Transmitted frames consist in 9999 QPSK symbols. The frame length has been chosen large enough in this example in order to stress the similarities between the behavior of the SMCRB and the bit-error rate (BER). Fig. 10.3 represents the BER versus the E_b/N_0-ratio when the turbo decoder is perfectly synchronized. Fig. 10.4 represents the SMCRB when the vector of soft information is made up with the extrinsic messages delivered by the turbo decoder. Note that the higher curve in Fig. 10.4 represents the SMCRB before entering the turbo decoder i.e. when all the extrinsic log-likelihood ratios are equal to zero. As mentioned above, the SMCRB then exactly corresponds to the NDA CRB.

Looking at Fig. 10.3 we may distinguish between three main regions: the non-convergence region ($E_b/N_0 < -0.5$dB), the water-fall region (-0.5dB $< E_b/N_0 <$ 0.5dB) and the convergence region ($E_b/N_0 >$ 0.5dB). In the non-convergence region, there is no improvement in the BER which remains stuck around 0.5. In the waterfall region, convergence occurs although very slowly. Finally, in the convergence region the BER floor is reached after only a few iterations. Similar regions can be observed in the SMCRB behavior in Fig. 10.4. For $E_b/N_0 < -0.5$dB, although the use of the code enables a performance improvement with respect to the NDA case, there is no improvement through the iterations. In the waterfall region (-0.5dB $< E_b/N_0 <$ 0.5dB), although the convergence is very slow, the SMCRB gets closer at each iteration to the MCRB. Finally in the convergence region ($E_b/N_0 >$ 0.5dB), the SMCRB reaches the MCRB after only a few iterations. This observation is quite important since it tends to show that iterative synchronization methods, which usually operate in the convergence region, should concentrate their effort only during the few first iterations.

10.2 Performance of Continuous-parameter Synchronizers

In this section we illustrate the performance of the SDD synchronizers derived in chapter 7. We will limit our considerations to the synchronization of the carrier phase offset and the timing offset. The channel amplitude and the carrier frequency offset will be assumed to be known. We consider the following synchronizers:

- *The conventional NDA synchronizers* [8, 9] : These estimators derive from approximations of the ML criterion and operate in a NDA mode. They do therefore not take into account the soft information available in the turbo receiver and their performance are bounded by the NDA CRB. In the sequel, we will consider the *Viterbi&Viterbi* (V&V) synchronizer [56] for carrier phase estimation and the *Oerder&Meyr* (O&M) synchronizer [55] for timing offset estimation (see section 7.2.2).
- *The EM synchronizer* : The CA synchronizer based on the EM algorithm is described in Algorithm 1 (see section 7.4). This synchronizer will be referred to as EM in the sequel.
- *The SD synchronizer* : The CA synchronizer based on the SD algorithm is described in Algorithm 3 (see section 7.5.1). The notation SD will be used to refer to it in the sequel. Regarding the choice of the step factor, we only consider the case of a constant step-size. Its value will be fixed by trial and error.
- *The SP synchronizer* : The CA synchronizer based on the SP algorithm is described in Algorithm 7. We will consider different implementation of the SP synchronizer:
  - *EM implementation :* The search of the maximum of $p^{(n)}(\mathbf{r}|\mathbf{b})$ is performed via an EM algorithm. This synchronizer is described in Algorithm 8 (see section 7.7.5). It will be denoted by SP-EM(n), where n represents the number of EM iterations performed at each SP iteration. Note that SP-EM(1) corresponds to the EM-synchronizer approximation described in Algorithm 2.
  - *SD implementation :* The search of the maximum of $p^{(n)}(\mathbf{r}|\mathbf{b})$ is performed via a SD algorithm (see Algorithm 9 in section 7.7.6). This synchronizer will be denoted by SP-SD(n), where n represents the number of SD iterations performed at each SP iteration. SP-SD(1) corresponds to the SD-synchronizer approximation described in Algorithm 4.
  - *NR implementation :* The maximum of $p^{(n)}(\mathbf{r}|\mathbf{b})$ is computed by means of the NR algorithm (see Algorithm 10 in section 7.7.6). This synchronizer will be referred to as SP-NR(n), where n represents the number of NR iterations performed at each SP iteration. SP-NR(1) corresponds to the NR-synchronizer approximation described in Algorithm 6.

The synchronizer based on the NR algorithm (see Algorithm 5) is too complex to implement and will not be considered in the sequel. Also, the ECM synchronizer described in section 7.4.2 will not be considered hereafter due to the limited number of synchronization parameters to estimate.

The performance of the synchronizers will be assessed in terms of frame-error rate (FER), bit-error rate (BER) and estimation mean square error (MSE). Note that these measures are not equivalent: two synchronization algorithms can for example lead to different BERs but exhibit the same FER. The ultimate performance criterion is of course the FER i.e. the portion of packet which cannot be recovered by the system. However, other criteria as the BER and the MSE can give interesting insights into the behavior of the proposed synchronization algorithms.

The remainder of this section is organized as follows. In a first subsection, we illustrate the speed of convergence of the SDD synchronizers. Then, we illustrate the ultimate performance in terms of MSE achievable by the proposed synchronizers. In a third subsection, we consider the acquisition range of the considered synchronization algorithms i.e. their sensibility to the choice of the initial value. Finally, we illustrate the performance achieved by different kind of receivers when they are synchronized by means of the proposed SDD synchronizers.

10.2.1 Synchronizer Performance: Speed of Convergence

In this subsection, we investigate the speed of convergence of the iterative synchronizers proposed in chapter 7. We first focus on the lower bound on the speed of convergence of the EM synchronizer derived in section 9.2. We show that the proposed lower bound is tight and enables therefore to predict the speed of convergence of the EM synchronizer by comparing the CRB and the MCRB associated to the estimation problem. In a second part, we compare the speed of convergence of the EM synchronizer with the one of the SD and NR synchronizers. For the sake of conciseness, we only consider carrier phase offset estimation. However, the same kind of conclusions can be drawn for timing estimation.

Relation between the speed of convergence of the EM synchronizer and the ratio CRB/MCRB

We illustrate that the speed of convergence of the EM synchronizer may be well-predicted by using (9.30), i.e.

$$E_{\mathbf{r}|b^\star}[\log \Delta^{(n)}] \geq n \log(1 - \text{MCRB}(b^\star)\text{CRB}^{-1}(b^\star)) + E_{\mathbf{r}|b^\star}[\log \Delta^{(0)}], \quad (10.4)$$

where $\Delta^{(n)} = \|\hat{b}^{(n)} - \hat{b}_{ML}\|$, $b^\star$ is the actual parameter value and n is the number of performed iterations. We then show that the speed of convergence of the EM synchronizer strongly depends on the setup of the considered communication system. In particular, we emphasize the effect of the constellation size, the signal-to-noise-ratio and the quantity of symbol a priori information.

Let us first focus on the effect of the SNR and the constellation size. Fig. 10.5 represents the CRB and the MCRB versus the E_s/N_0-ratio for the transmission of 500 uncoded 8-PSK, QPSK and BPSK symbols. Fig. 10.6 represents the lower bound on the mean rate associated to these CRBs[1] (see chapter 9). We note that, for a given E_s/N_0-ratio, the CRB increases with the constellation size. The same effect is observed in Fig. 10.6: the lower bound on the mean converge rate is all the higher as the constellation size increases. This means that the EM synchronizer will have a slower converge if applied to the synchronization of a 8-PSK transmission than if applied to the synchronization of a BPSK transmission. For a given constellation size, we see that the CRB gets closer to the MCRB when the E_s/N_0-ratio increases. As a consequence, we see from Fig. 10.6 that the lower bound on the mean convergence rate decreases when the E_s/N_0-ratio increases. According to this result, we may expect the EM algorithm to have a faster speed of convergence when the operating SNR is higher.

Fig. 10.7 and 10.8 compare Monte-Carlo simulation results with the bound on the speed of converge proposed in this thesis. Fig. 10.7 represents the effect of the constellation size whereas Fig. 10.8 illustrates the effect of the SNR. We may notice in both cases that the simulation points almost perfectly stick to the lower bounds. The proposed lower bound is therefore tight and may be used to predict the convergence behavior of the EM synchronizer by simply considering the gap between the CRB and the MCRB.

[1] i.e. $\log(1 - \text{MCRB}/\text{CRB})$.

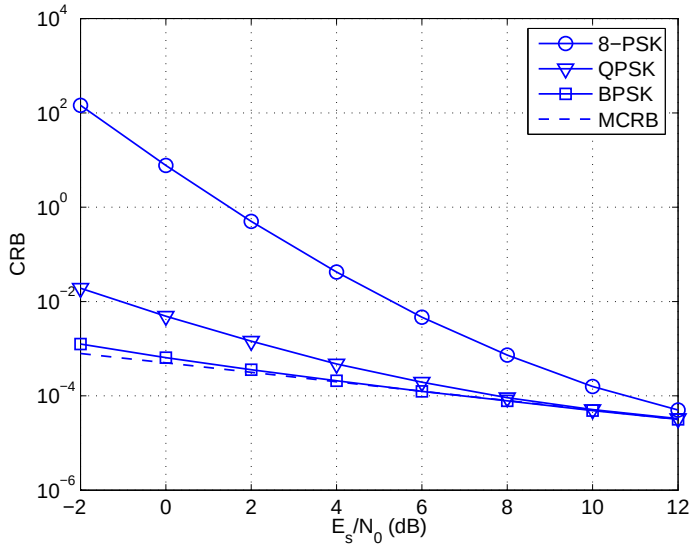

Figure 10.5 CRB versus E_s/N_0 for an uncoded transmission. The frames are made up with 500 symbols. The curves are plotted for different constellation sizes.

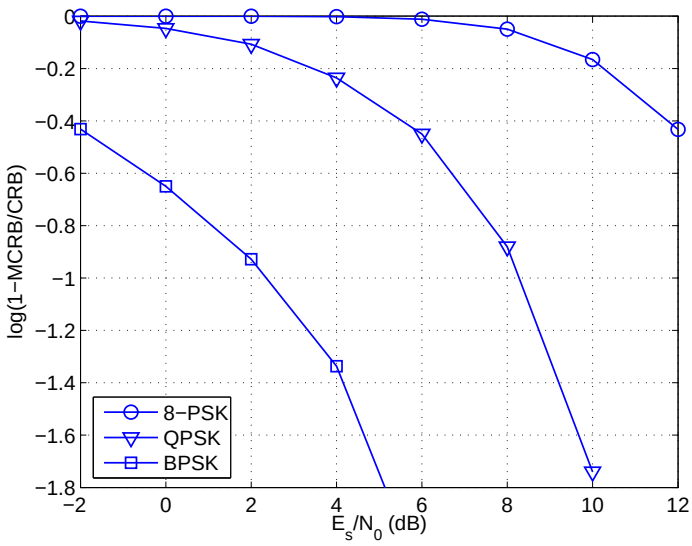

Figure 10.6 $\log(1 - \text{MCRB}/\text{CRB})$ versus E_s/N_0 for an uncoded transmission. The frames are made up with 500 symbols. The curves are plotted for different constellation sizes.

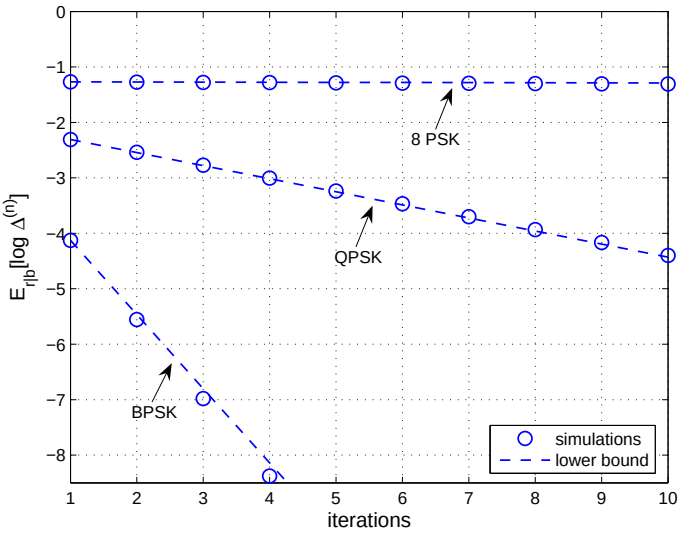

Figure 10.7 Comparison of $E_{\mathbf{r}|b}[\log \Delta^{(n)}]$ computed by Monte-Carlo simulations with the proposed lower bound. The E_s/N_0-ratio is set to 4 dB. The curves are plotted for different constellation sizes.

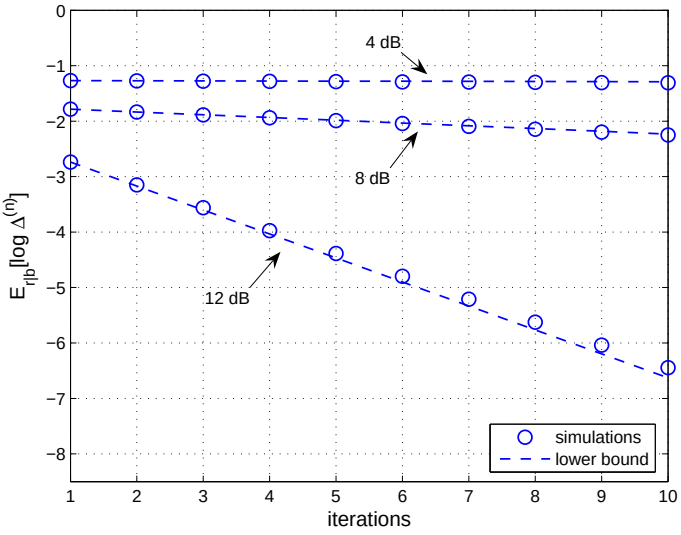

Figure 10.8 Comparison of $E_{\mathbf{r}|b}[\log \Delta^{(n)}]$ computed by Monte-Carlo simulations with the proposed lower bound. We consider a 8-PSK modulation. The curves are plotted for different SNRs.

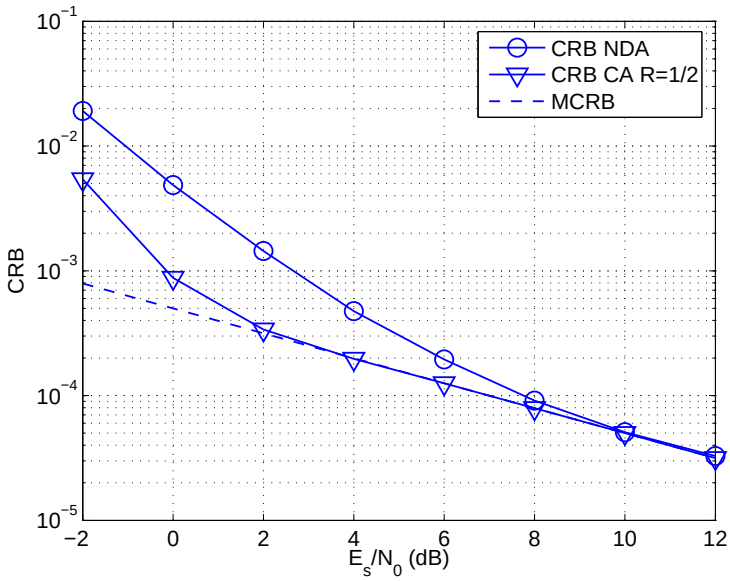

Figure 10.9 CRB versus the E_s/N_0-ratio for a coded and an uncoded transmission.

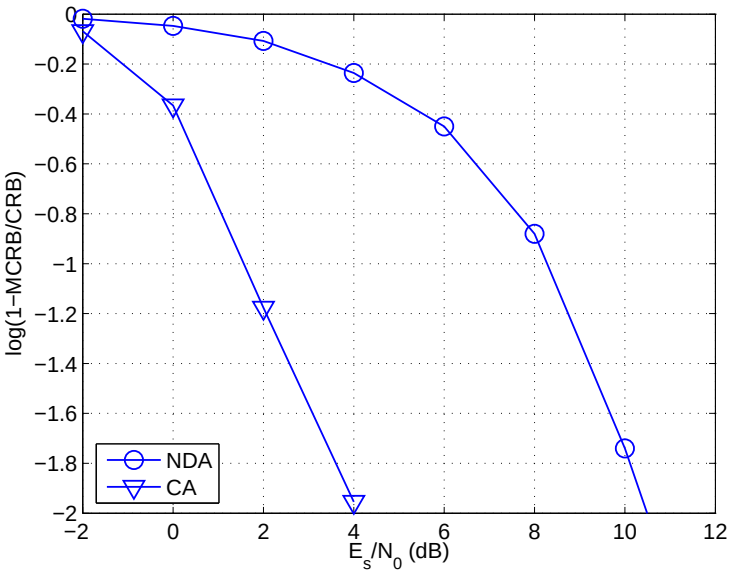

Figure 10.10 $\log(1 - \text{MCRB}/\text{CRB})$ versus E_s/N_0 for a coded and an uncoded transmission.

As a last example of the utility of the proposed bound, we consider the effect of the amount of a priori information about the transmitted sequence. Fig. 10.9 represents the CRB respectively associated to a coded and an uncoded transmission. We have considered the transmission of 500 QPSK symbols and a rate-1/2 convolutional code with constraint length equal to 3. The code structure provides some a priori information about the transmitted sequence and the associated CRB is always lower than the NDA CRB. As a consequence, we may notice in Fig. 10.10 that the mean rate of convergence of the EM synchronizer is always lower, i.e. implies faster convergence for coded transmissions than for uncoded ones. Let us moreover mention that the operating point of practical coded transmissions is usually such that BER $< 10^{-3}$. For such a low BER, we have that CRB$\simeq$MCRB (see [12]). Therefore, the EM synchronizer should always exhibit a quite fast speed of convergence when it computes the CA ML estimate. On the contrary, the application of the EM algorithm for the computation of the NDA ML solution may lead to a very poor speed of convergence.

Comparison of the Synchronizer Speed of Convergence

We will compare the speed of convergence of the EM synchronizer with the speed of convergence of the gradient-based synchronizers, namely the SD and the NR synchronizers.

We consider the transmission of 1000 8-PSK symbols. The E_s/N_0-ratio is set to 5.77 dB. All the synchronization parameters but the carrier phase offset are assumed to be known. The likelihood function to maximize is built from a vector of soft information, i.e.

$$p(\mathbf{r}|\mathbf{b},\mathbf{e}) = \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b})\, p(\mathbf{a}|\mathbf{e}). \tag{10.5}$$

In the same way as in section 10.1.1, we assume that the soft information is delivered under the form of log-likelihood ratios on the coded bits. We consider that $p(\mathbf{a}|\mathbf{e}) = \prod_k p(a_k|e_k)$ is computed according to (10.1) and (10.2). The quantity of a priori information is therefore a direct function of the mutual information between $x_{k,l}$ and $e_{k,l}$ (see section 10.1.1). We consider the cases where $I(x_{k,l}, e_{k,l})$ is equal to 0, 0.2, 0.8 and 1. The case $I(x_{k,l}, e_{k,l}) = 0$ corresponds to the NDA case i.e. no a priori information is available about the transmitted sequence. The case $I(x_{k,l}, e_{k,l}) = 1$ corresponds to the DA case i.e. we have a perfect knowledge of the transmitted sequence. $I(x_{k,l}, e_{k,l}) = 0.2$

and $I(x_{k,l}, e_{k,l}) = 0.8$ correspond to intermediate cases.

Fig. 10.11 to 10.14 illustrate the MSE achieved by the EM, SD and NR synchronizers versus the number of iterations for different values of $I(x_{k,l}, e_{k,l})$. Note that the MSE is averaged over the distribution of the soft information vector $\mathbf{e}$. The actual carrier phase offset is equal to 0 degree and all the synchronizers are initialized at $\hat{\theta}^{(0)} = 11.25°$. The NR synchronizer is run with a step factor $\alpha = 1$. From these figures, we can make the following comments:

- The EM synchronizer has a very low speed of convergence (see Fig. 10.11 and 10.12) when we have only a little information about the transmitted sequence at our disposal. On the contrary, when we may benefit from a lot of information about the transmitted sequence, the EM synchronizer exhibits a fast speed of convergence (see Fig. 10.13 and 10.14). This result is in accordance with the convergence behavior of the EM synchronizer mentioned above: the EM algorithm converges all the faster as the quantity of a priori information about the transmitted sequence increases.

- The speed of convergence of the SD synchronizer strongly depends on the choice of the step-factor α. In our simulations, we consider $\alpha = 10^{-3}$ and $\alpha = 10^{-4}$. We see that the choice of a "good" step-factor depends on the quantity of information we have about the transmitted sequence. On the one hand, when $I(x_{k,l}, e_{k,l})$ is small, $\alpha = 10^{-3}$ enables a reasonably fast convergence. On the other hand, when $I(x_{k,l}, e_{k,l})$ is larger, this value of the step factor prevents the convergence of the SD synchronizer. In the same way, $\alpha = 10^{-4}$ leads to a fast convergence for large values of $I(x_{k,l}, e_{k,l})$ but slows down the convergence for smaller values of $I(x_{k,l}, e_{k,l})$.

- The NR synchronizer exhibits a fast speed of convergence whatever the value of the mutual information $I(x_{k,l}, e_{k,l})$ is. This is due to the fact that, unlike the EM synchronizer, the speed of convergence of the NR synchronizer is not significantly affected by the amount of information available about the transmitted sequence. We see that, in the considered case, its convergence is always faster (or equal for $I(x_{k,l}, e_{k,l}) = 1$) than the convergence of the EM and SD synchronizers. Note that we have implemented in our simulations the method proposed in section 7.5.4 to detect the convergence to a minimum : if the second derivative is posi-

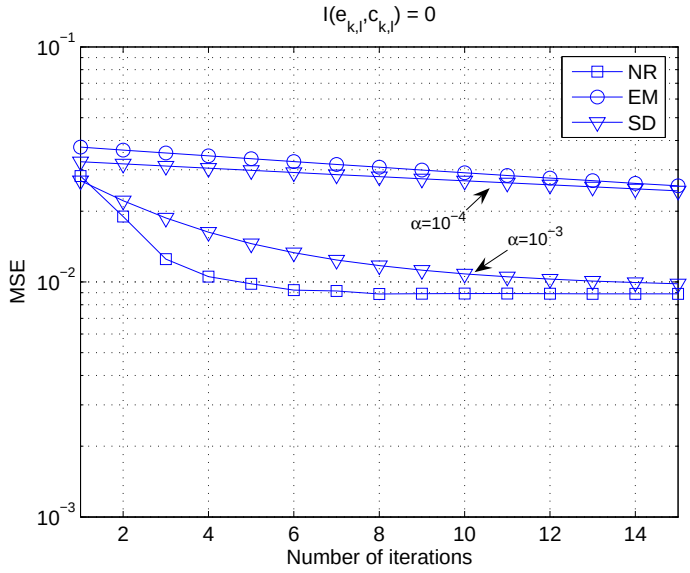

Figure 10.11 MSE achieved by different carrier phase offset synchronizers versus the number of iterations for $I(e_{k,l}, x_{k,l}) = 0$. The actual phase offset is 0° and the synchronizers are initiated at $\hat{\theta}^{(0)} = 11.25^\circ$. The constellation is 8-PSK.

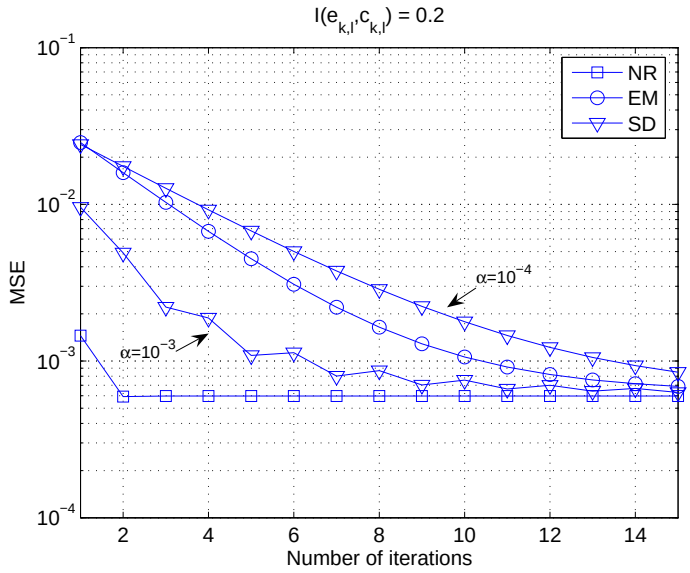

Figure 10.12 MSE achieved by different carrier phase offset synchronizers versus the number of iterations for $I(e_{k,l}, x_{k,l}) = 0.2$. The actual phase offset is 0° and the synchronizers are initiated at $\hat{\theta}^{(0)} = 11.25^\circ$. The constellation is 8-PSK.

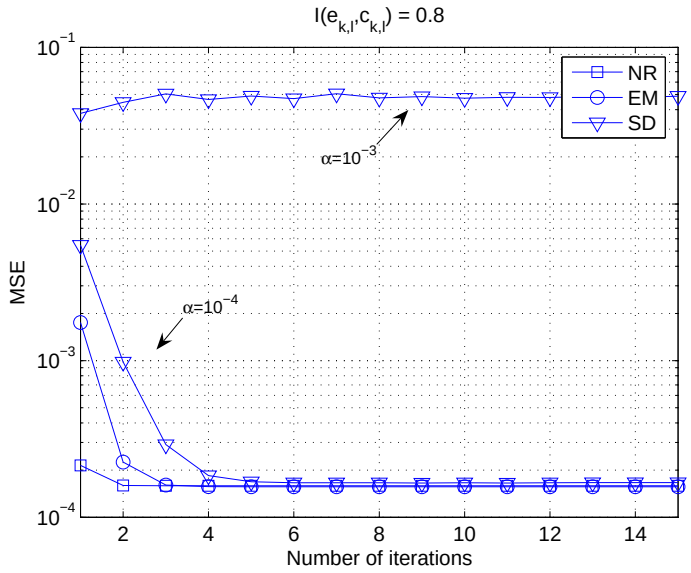

Figure 10.13 MSE achieved by different carrier phase offset synchronizers versus the number of iterations for $I(e_{k,l}, x_{k,l}) = 0.8$. The actual phase offset is 0° and the synchronizers are initiated at $\hat{\theta}^{(0)} = 11.25^\circ$. The constellation is 8-PSK.

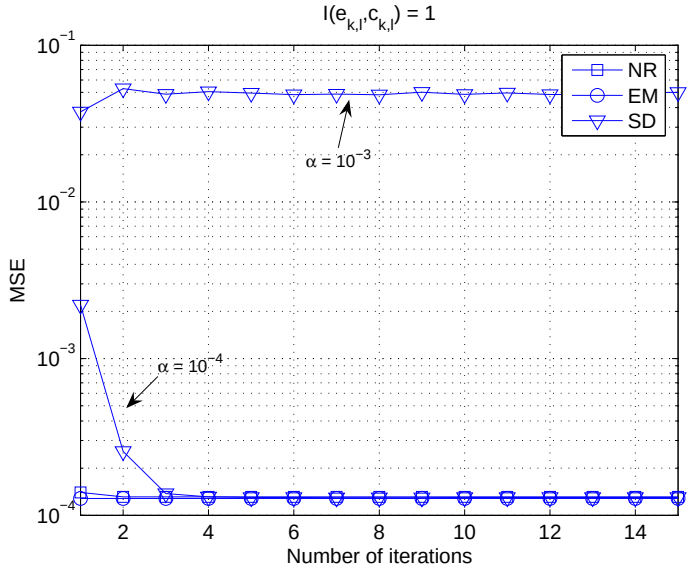

Figure 10.14 MSE achieved by different carrier phase offset synchronizers versus the number of iterations for $I(e_{k,l}, x_{k,l}) = 1$. The actual phase offset is 0° and the synchronizers are initiated at $\hat{\theta}^{(0)} = 11.25^\circ$. The constellation is 8-PSK.

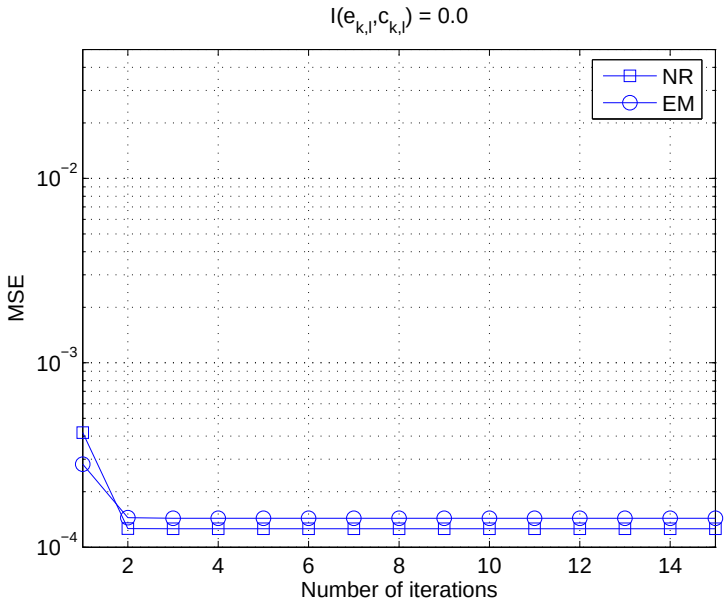

Figure 10.15 MSE achieved by different carrier phase offset synchronizers versus the number of iterations for $I(e_{k,l}, x_{k,l}) = 0$. The actual phase offset is 0° and the synchronizers are initiated at $\hat{\theta}^{(0)} = 45^\circ$. The constellation is BPSK.

tive, we conclude that the NR synchronizer converges to a minimum and 11.25° are subtracted from the estimate. This way, the new estimate is in the "attraction region" of a maximum.

Note that the setup considered in the previous simulation is quite difficult for the EM synchronizer. Indeed, we have seen in the previous section that the convergence of the EM synchronizer (in the case of carrier phase estimation) is slow for high-order PSK constellations. In Fig. 10.15, we emphasize that the converge of the EM synchronizer may be as fast fast as the one of the NR synchronizer depending on the considered case. In this figure, we consider BPSK transmission. The initial estimate is set to 45° and $I(x_{k,l}, e_{k,l}) = 0$. The other parameters are the same as above. We note that in this case, the EM and the NR synchronizers have the same speed of convergence although we do not have any information about the transmitted sequence.

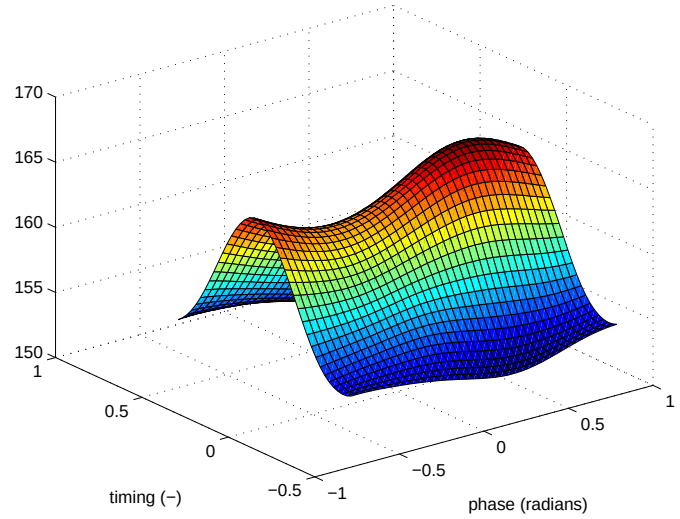

Figure 10.16 3-dimensional representation of the log-likelihood function. We consider carrier phase and timing estimation. The frames consist in 500 QPSK symbols and $E_s/N_0 = 0$ dB.

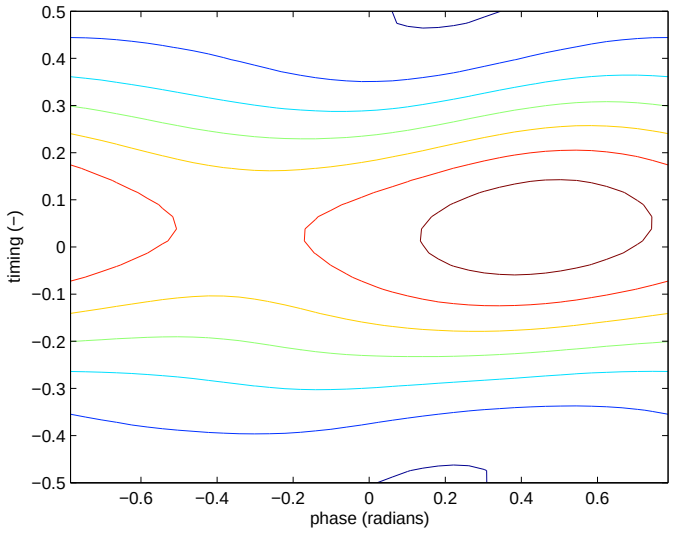

Figure 10.17 Contour-representation of the log-likelihood function. We consider carrier phase and timing estimation. The frames consist in 500 QPSK symbols and $E_s/N_0 = 0$ dB.

The Synchronizer Trajectory

In the previous example, we have considered a one-dimensional problem. We now consider the joint estimation of the carrier phase offset and the timing offset. In this two-dimensional problem, it is interesting to study the trajectory of the synchronizers. This trajectory will actually be a function of the speed of convergence of the synchronizer along each dimension: if the synchronizer converges as fast in each dimension, the trajectory should be close to a straight line; on the contrary, if the synchronizer speed of converge is faster along one dimension, the trajectory will be curved.

We illustrate these considerations on the following example. We consider the transmission of 500 uncoded QPSK symbols. The E_b/N_0-ratio is set to 0 dB. The roll-off factor is equal to 0.2. Fig. 10.16 and 10.17 are respectively a two-dimensional and a contour representation of the log-likelihood function to maximize. Fig. 10.18 to 10.20 illustrate the trajectories of the EM, SD and NR synchronizers. In these figures, each arrow represents the direction and the magnitude of the gradient vector at the considered point. Note that the step factor of the NR synchronizer has been set to 0.1 in order to better visualized its trajectory. The synchronizers are initialized at $\theta = 0.6$ radians and $\tau = 0.125$. From these figures, we can make the following comments:

- The trajectory of the EM synchronizer is curved: we see that the EM algorithm converges faster in the direction of the timing than in the direction of the carrier phase. This behavior may be understood via the lower bound derived in chapter 9. Indeed, the latter bound states that the speed of convergence of the EM synchronizer is all the slower as the gap between the CRB and the MCRB is large. Now, in the considered case, the CRB associated to the carrier phase is quite far from the corresponding MCRB whereas the CRB associated to timing estimation is close to the corresponding MCRB.
- Like the EM synchronizer, the trajectory of the SD synchronizer is also curved. The justification is however not the same as for the EM synchronizer. The speed of convergence of the SD synchronizer depends on the ratio of the phase and timing second derivatives [47]. In the considered case, we see that the second derivative is more negative in the direction of the timing than in the direction of the carrier phase. As a consequence, the convergence of the SD algorithm is slower in the direction of the car-

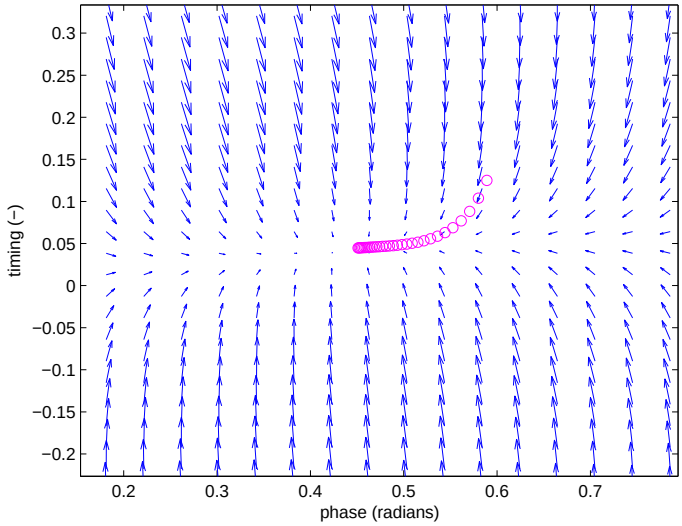

Figure 10.18 Instance of trajectory of an EM synchronizer. The circles represent the sequence of estimates computed by the EM synchronizer. The arrows represent the direction and the amplitude of the gradient at the considered point.

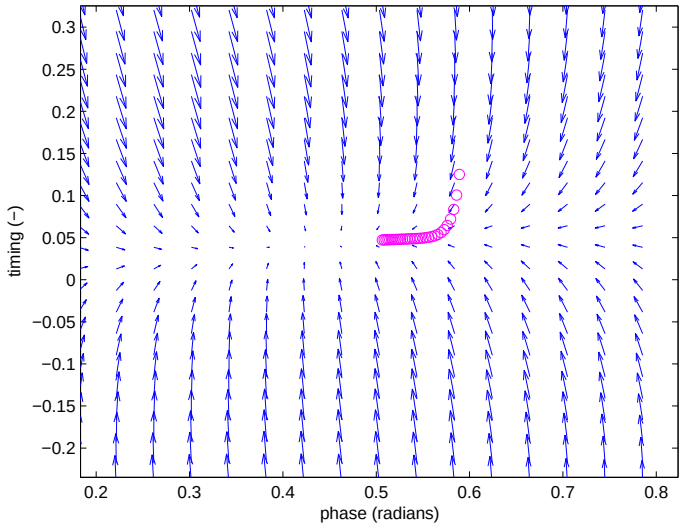

Figure 10.19 Instance of trajectory of a SD synchronizer. The circles represent the sequence of estimates computed by the SD synchronizer. The arrows represent the direction and the amplitude of the gradient at the considered point.

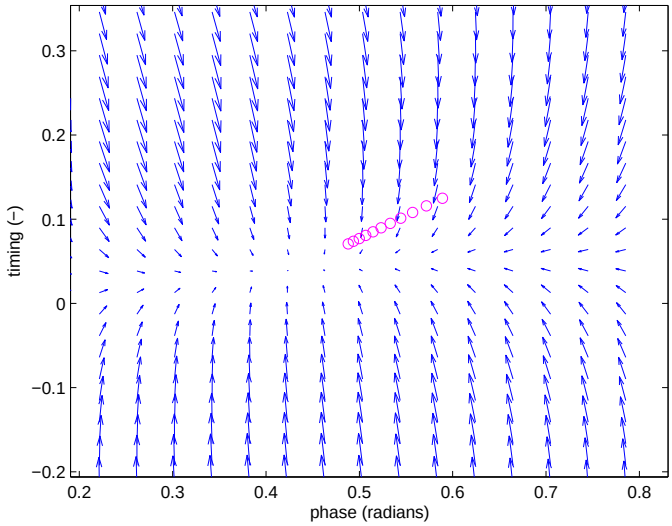

Figure 10.20 Instance of trajectory of a NR synchronizer. The circles represent the sequence of estimates computed by the NR synchronizer. The arrows represent the direction and the amplitude of the gradient at the considered point.

rier phase offset.

Note that since the factors affecting the convergence of the EM and the SD synchronizers are not the same, these synchronizers do not necessarily exhibit their slowest speed of convergence along the same dimension as in the considered example.

- The NR synchronizer trajectory is a straight line. In fact, the NR synchronizer gets rid of the problem affecting the SD synchronizer by "preconditioning" the gradient by the inverse of the Hessian matrix. In the this way, the ratio between the second derivatives do no longer affect the speed of convergence of the synchronizer [47].

10.2.2 Synchronizer Performance: Ultimate Accuracy

In this section, we illustrate the MSE achievable by different synchronizers. In particular, we compare the performance of the proposed SDD synchronizers, when operating in either NDA or CA mode, with the performance of conventional NDA synchronizers. We distinguish between two cases: the conditional factor graph is cycle-free or is not. By conditional factor graph we mean the

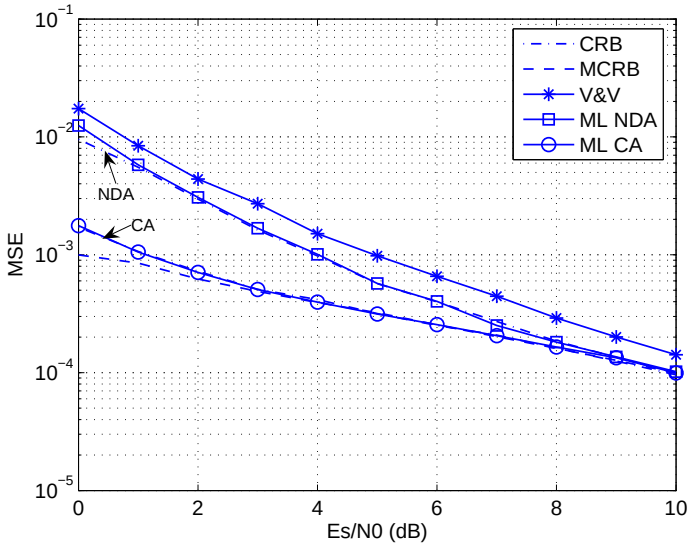

Figure 10.21 Phase estimation: MSE versus E_s/N_0-ratio for a convolutionally-coded transmission. The frames consist in Gray-mapped 500 QPSK symbols.

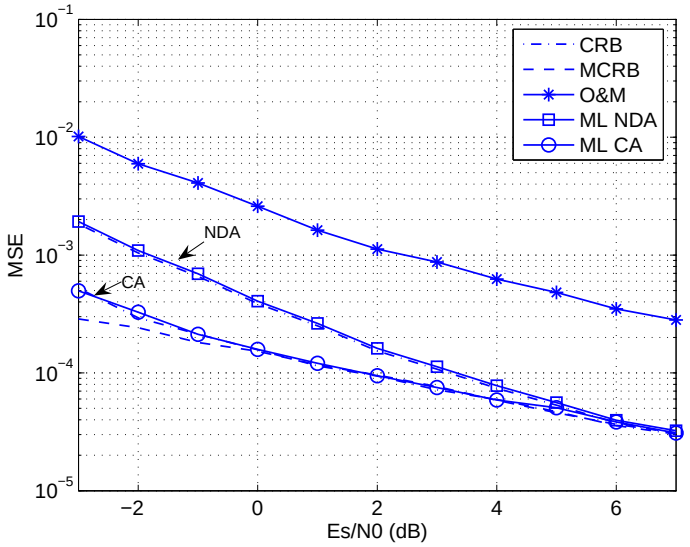

Figure 10.22 Timing estimation: MSE versus E_s/N_0-ratio for a convolutionally-coded transmission. The frames consist in 1000 BPSK symbols and the roll-off factor is equal 0.1.

factor graph of the system which assumes the synchronization parameters to be known.

Cycle-free Conditional Factor Graph

When the conditional factor graph has no cycles, the a posteriori marginals $p(a_k|\mathbf{r},\mathbf{b})$ required to implement the EM and the SD synchronizers may be exactly computed (This is, for example, the case for a BPSK convolutionally-coded transmission). Therefore, if these algorithms are properly initialized, they both will reach the ML estimate after a sufficient number of iterations[2]. Note that the ML estimator is unbiased and its covariance is the inverse of the Fisher information matrix as long as the number of observations is large enough (see chapter 3). As a consequence, the EM and the SD synchronizers should reach the CRB for a sufficiently-large frame length. Note also that the EM and the SD synchronizers may operate in either CA or NDA mode depending on whether they take or not the code structure into account.

Fig. 10.21 represents the MSE achieved by different synchronizers for carrier-phase estimation in the case of a convolutionally-coded transmission. We consider a rate-1/2 length-3 non-systematic convolutional code with maximum free-distance. The transmitted frames consist in 500 QPSK symbols. We use a Gray mapping. The other synchronization parameters are assumed to be known. The MCRB and the CRBs associated to the CA and the NDA synchronization modes are also represented. The performance of the EM and the SD synchronizers is similar and is only represented by one curve. The curve labeled "ML CA" represents the MSE of the SDD synchronizers when operating in a CA synchronization mode. In the same way, the curve labeled "ML NDA" represents the MSE of the SDD synchronizers when operating in a NDA synchronization mode. Note that the SDD synchronizers are initialized in the "attraction region" of the global maximum of the likelihood function and are run until convergence. We see that the CA (resp. NDA) SDD synchronizers enable to reach the CA (resp. NDA) CRB. This is due to the fact that the SDD synchronizers deliver the ML estimate when properly initialized. The curve labeled "V&V" illustrates the performance of the conventional NDA Viterbi&Viterbi carrier phase synchronizer (see section 7.2.2). The MSE achieved by the Viterbi&Viterbi synchronizer does not exactly stick the NDA CRB. This is due to the fact that the Viterbi&Viterbi synchronizer

[2] As long as the step-factor α is well-chosen.

does not compute the true ML solution but only an approximation of it. The reason of the gap between the MSE achieved by the CA synchronizers and the conventional synchronizer is therefore twofold: the Viterbi&Viterbi operates in a NDA mode and it does not compute the actual NDA ML solution. Note that at low SNR, the NDA operating mode is the main source of degradation.

Fig. 10.22 represents the MSE achieved by different synchronizers for timing-offset estimation in the case of a convolutionally-coded transmission. The MCRB and the CRBs associated to the CA and the NDA synchronization modes are also represented. The code is a rate-1/2 length-3 non-recursive code with maximum free-distance. The transmitted frames consist in 1000 BPSK symbols. The roll-off factor has been set to 0.1. The other synchronization parameters are assumed to be known. As for the carrier phase offset, we note that the CA (resp. NDA) SDD synchronizers enable to reach the CA (resp. NDA) CRB. On the contrary, the conventional NDA Oerder&Meyr (O&M) timing synchronizer leads to important degradation with respect to the NDA CRB. This is due to the approximations made in the design of the O&M synchronizer (see section 7.2.2). We see therefore that the gap between the CA synchronizer and the O&M synchronizer performance is the combination of two effects: the O&M losses estimation accuracy due to its NDA nature and because it does not compute the actual NDA ML solution. Note that the performance of the O&M synchronizer is quite sensitive to the value of the roll-off factor: smaller roll-off factors leads to larger degradation of the performance. This is why we observe a so large gap between the MSE achieved by the O&M and the NDA CRB.

As a conclusion, we see that the SDD synchronizers enable to reach the NDA or the CA CRB (depending on the operating mode) as long as they are properly initialized and a sufficient number of iterations is performed. The improvement brought by the proposed CA synchronizers with respect to conventional synchronizers is twofold: 1) they operate in a CA mode i.e. improve the performance by taking benefit from the code structure; 2) they compute the true ML solution and avoid therefore the degradation due to the approximations typically made in the design of conventional synchronizers.

Conditional Factor Graph with Cycles

We now consider the performance of the CA SDD synchronizers when the conditional factor graph contains cycles. In this case, the computation of

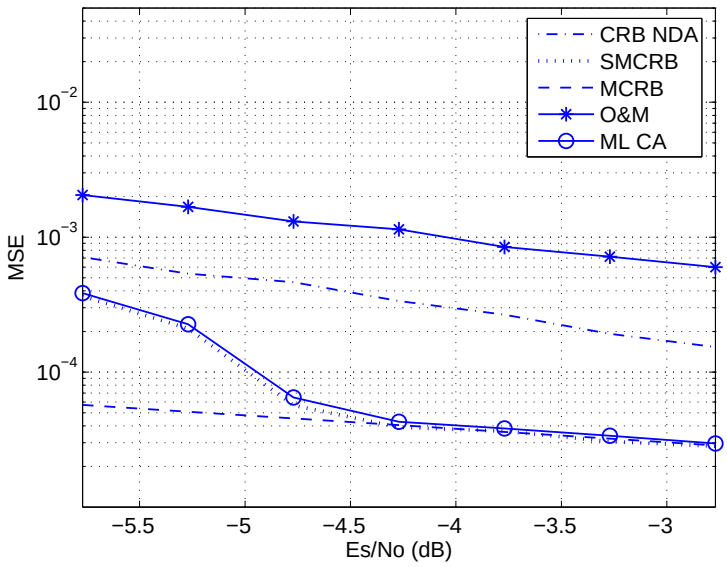

Figure 10.23 Timing estimation: MSE versus E_s/N_0-ratio for a turbo-coded transmission. The turbo encoder is made up with two rate-1/2 recursive systematic convolutional codes with octal generators $(21, 37)_8$. The frame length consists in 9999 BPSK symbols and the roll-off factor is set to 0.1.

marginals $p(a_k|\mathbf{r}, \mathbf{b})$ (or rather an approximation of them) requires to run an iterative SP algorithm (see chapter 5). The implementation of the EM and SD synchronizers is therefore far too complex in practice and will not be considered here. Instead, we consider the SP synchronizers (SP-EM, SP-SD or SP-NR) which only slightly increase the complexity of the overall receiver. As in the previous section, the SP synchronizers are initialized in the attraction region of the global maximum of the modified likelihood function and run until convergence. As a consequence, they all lead to the same performance and only one curve will be represented.

Fig. 10.23 represents the MSE achieved by different synchronizers in the case of timing offset estimation for a turbo-coded transmission. The MCRB and the NDA CRB are also represented. Moreover we have plotted the SMCRB computed from the extrinsic LLRs delivered by the perfectly-synchronized turbo decoder after convergence. We consider a rate-1/3 turbo code made up with two rate-1/2 recursive systematic convolutional codes

with octal generators $(21, 37)_8$. The frame length consists in 9999 BPSK symbols. The roll-off factor is set to 0.1. We may observe that the performance of the SP synchronizer perfectly sticks to the SMCRB. On the contrary the O&M is unable to reach the NDA CRB. As mentioned above, this is due to the approximations made in the design of the O&M synchronizer. We see that, in this case, the implementation of a CA synchronizer enables to greatly improve the estimation quality with respect to the conventional NDA synchronizer.

10.2.3 Synchronizer Performance: Acquisition Range

In this subsection, we emphasize that the convergence behavior of the proposed iterative synchronizers depends on the initial estimate. In particular we consider the acquisition range of the synchronizers i.e. the range of initialization values for which the synchronizers still converge to the ML solution.

The EM and the gradient synchronizers proposed in chapter 7 are "hill-climbing" algorithms i.e. algorithms converging to the top of the "local" hill[3]. If the algorithms are initiated close to the global maximum, the synchronizers will deliver the ML estimate; otherwise they may deliver another value. The study of the structure of the function to maximize can therefore give useful insights into the convergence behavior of the iterative synchronizers. Fig. 10.24 represents a particular realization of the log-likelihood function $\log p(\mathbf{r}|\theta)$ in both the cases of a coded and an uncoded BPSK transmission. We have considered two rate-1/2 non-systematic convolutional codes with constraint length equal to 3 and 5 respectively. The actual carrier phase offset has been set to 0 degrees. We may observe a 180-degree periodicity of the NDA log-likelihood function. This periodicity is actually due to the 180-degree symmetry of the BPSK constellation. On the contrary, we see that the use of the code enables to remove this periodicity: the CA log-likelihood function has only one global maximum in the range $[-180^\circ, 180^\circ]$. As a consequence, handling the CA likelihood function should allow to deal with the phase ambiguity which arises in NDA synchronizers. Nevertheless, we will see that the proposed SDD iterative synchronizers are unable to take benefit from the code structure to remove this ambiguity.

Fig. 10.25 represents the acquisition range of the SDD synchronizers

[3]Let us remind that the NR algorithm may also converge to minima.

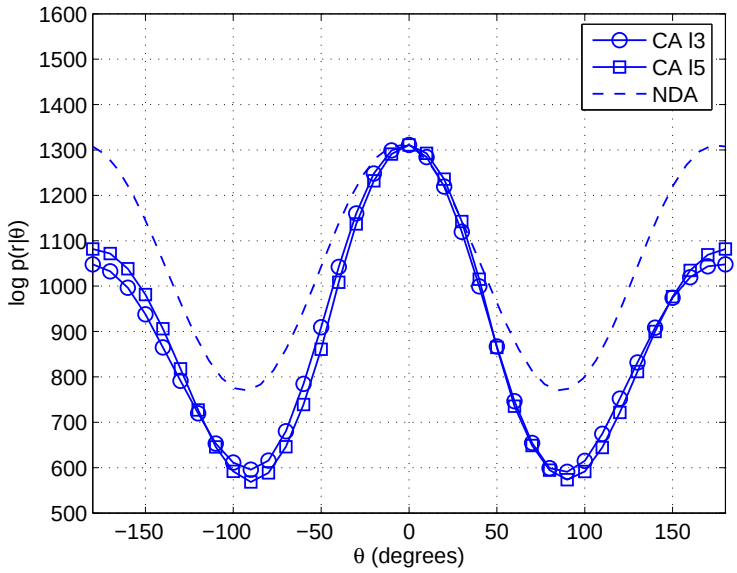

Figure 10.24 Log-likelihood function for uncoded and coded transmissions.

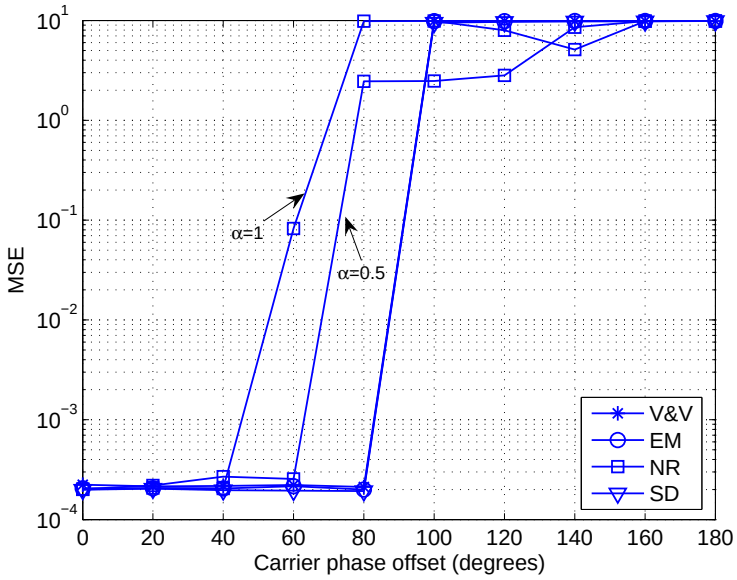

Figure 10.25 MSE versus the value of the carrier phase offset when $\hat{\theta}^{(0)} = 0$.

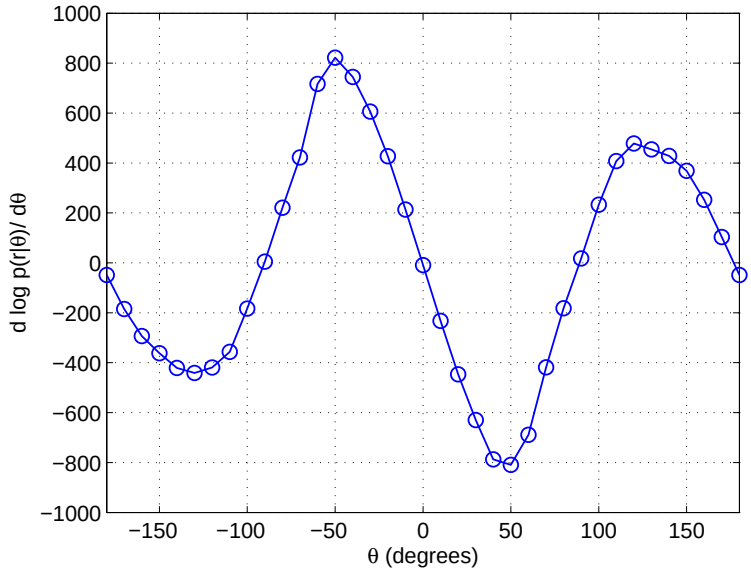

Figure 10.26 First derivative of the log-likelihood function for a convolutional code with constraint length equal to 3.

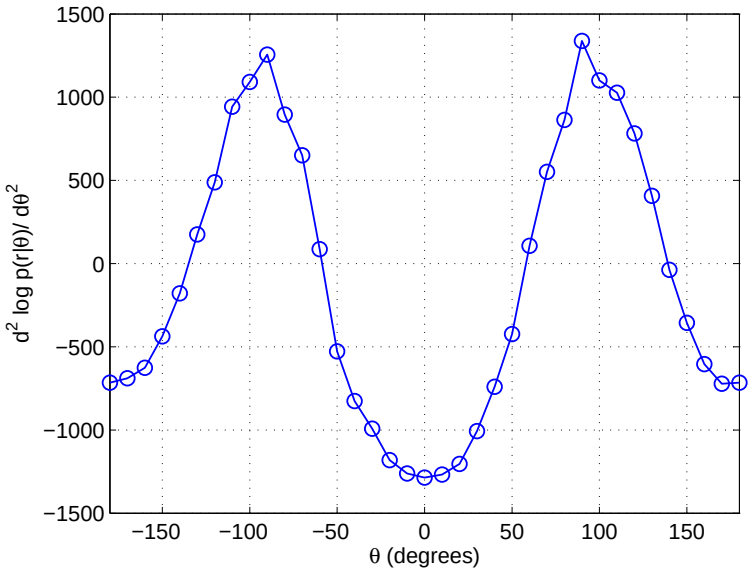

Figure 10.27 Second derivative of the log-likelihood function for a convolutional code with constraint length equal to 3.

i.e. the range of values of the carrier phase offset that the synchronizers are able to recover if initialized at $\hat{\theta}^{(0)} = 0°$. The MSE achieved by the Viterbi&Viterbi synchronizer is also represented. The results are only illustrated for $\theta \in [0°, 180°]$ but are also valid on the range $[-180°, 0°]$ due to the symmetry of the problem. We see that the Viterbi&Viterbi synchronizer suffers from the typical ambiguity of NDA synchronizers: it is unable to recover phase offsets larger than 90°. The EM and the SD CA synchronizers exhibit the same acquisition range as the Viterbi&Viterbi synchronizer. The NR synchronizer has a smaller acquisition range than the other synchronizers. We see therefore that the proposed SDD CA synchronizers are unable to increase the acquisition range. The behavior of the SD algorithm is easy to understand by looking at the first derivative of $\log p(\mathbf{r}|\theta)$ in Fig. 10.26. We see that the SD synchronizer has three fixed point, namely -180°, 0° and 180°. It is clear from this figure that the synchronizer will converge to 0° when it is initialized between -90° and 90°. Otherwise, it will converge to -180° or 180°. The limited acquisition range of the SD synchronizer is therefore due to the fact that gradient optimization methods may converge to either global or local maxima. The convergence behavior of the NR algorithm is even more critical: it may converge to a local minimum when the second derivative of $\log p(\mathbf{r}|\theta)$ is positive. From Fig. 10.27 we see that the acquisition range of the NR synchronizer is limited to the interval [-60°, 60°]. In Fig. 10.25 we even see that the NR synchronizer already diverges from $\theta = 40°$ when $\alpha = 1$. Indeed, when the second derivative is very small, the correction performed by the NR algorithm can be too large and the synchronizer therefore diverges. We see that imposing a step factor smaller than 1 enables to improve the robustness of the NR synchronizer. This improvement is however at the expense of a decrease of the speed of convergence.

As a conclusion, due to both the structure of the likelihood function and the nature of the considered optimization algorithms (i.e. they may converge to local maxima), we see that the proposed SDD CA synchronizers are (usually) not able to increase the acquisition range with respect to conventional NDA synchronizers. The NR synchronizer is the less robust since it also converges to local minima.

10.2.4 Synchronization of State-of-the-art Receivers

The properties of the SDD iterative synchronizers have been studied in section 10.2.1, 10.2.2 and 10.2.3. In this section, we will study the impact of these synchronization algorithms on the BER and the FER achieved by state-of-the-art receivers. Before we continue, let us mention that not all state-of-the-art receivers require CA synchronization. In some cases, conventional synchronization methods will just perform fine and will be sufficient for properly synchronizing the system. In other cases, the use of more powerful synchronizers are required. In practice, the choice of a proper synchronization method has therefore to be made on a case-by-case basis. In the sequel, we will choose examples which enable to show off the power and the relevance of the proposed CA algorithms. We consider the synchronization of three different kinds of receivers: a BCJR decoder, a turbo decoder and a turbo demodulator.

Convolutionally-coded Transmission

We consider a rate-1/3 non-systematic convolutional (NSC) encoder with generator polynomials $(25, 33, 37)_8$. The trellis of the code is terminated. The coded bits are mapped onto a QPSK constellation with Gray mapping. At the receiver, we use a BCJR decoder to recover the transmitted bits. Each frame is made up with 252 QPSK symbols. The roll-off factor is equal to 0.2. The carrier phase offset θ and the timing offset τ are assumed to be uniformly distributed respectively on the interval $[-\pi/4, \pi/4]$ and $[-0.5T, 0.5T]$, and they are changed for each new transmitted frame. The other synchronization parameters are assumed to be known.

Fig. 10.28 and 10.29 represent the FER and the BER achieved by the system when synchronized according to different approaches. We consider the conventional approach based on the Viterbi&Viterbi (V&V) [56] and the Oerder&Meyr (O&M) [55] synchronizers as well as the EM, SD, SP-EM, SP-SD SDD synchronizers. All the iterative synchronizers are initialized with a phase and a timing estimate computed by means of a V&V and an O&M synchronizer. Five iterations are performed. We assume that both the frame synchronization and the phase ambiguity problems are perfectly solved i.e. at each iteration the timing estimate $\hat{\tau}$ is constrained to be such

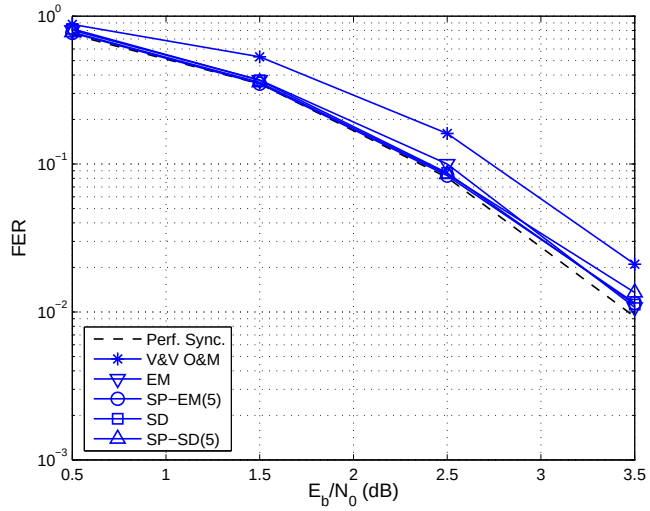

Figure 10.28 FER versus E_b/N_0 for a convolutionally-coded transmission synchronized according to different methods. We consider a rate-1/3 non-systematic convolutional (NSC) encoder with generator polynomials $(25, 33, 37)_8$. Each frame is made up with Gray-mapped 252 QPSK symbols. Phase and time synchronized are performed.

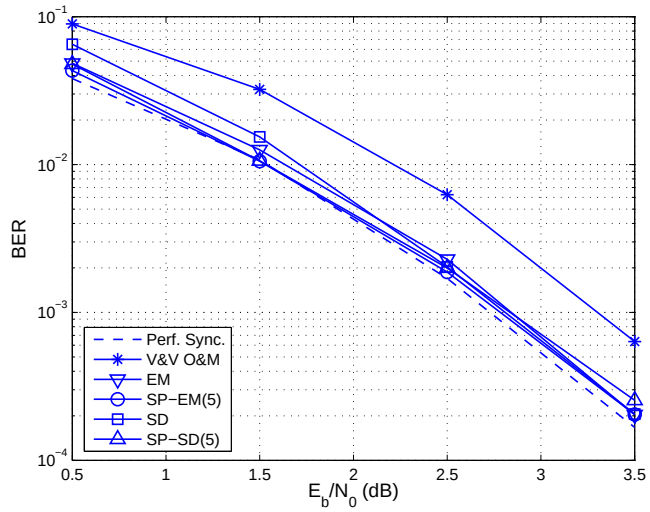

Figure 10.29 BER versus E_b/N_0 for a convolutionally-coded transmission synchronized according to different methods. We consider a rate-1/3 non-systematic convolutional (NSC) encoder with generator polynomials $(25, 33, 37)_8$. Each frame is made up with Gray-mapped 252 QPSK symbols. Phase and time synchronized are performed.

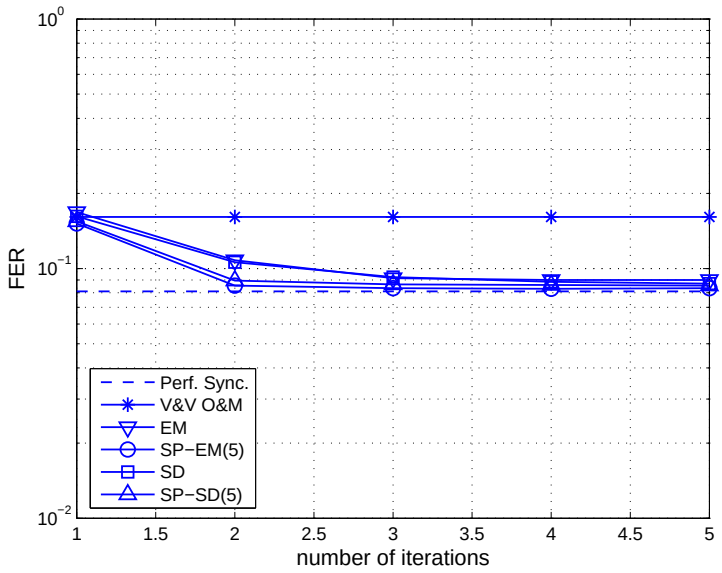

Figure 10.30 FER versus the number of iterations at $E_b/N_0 = 2.5$ dB.

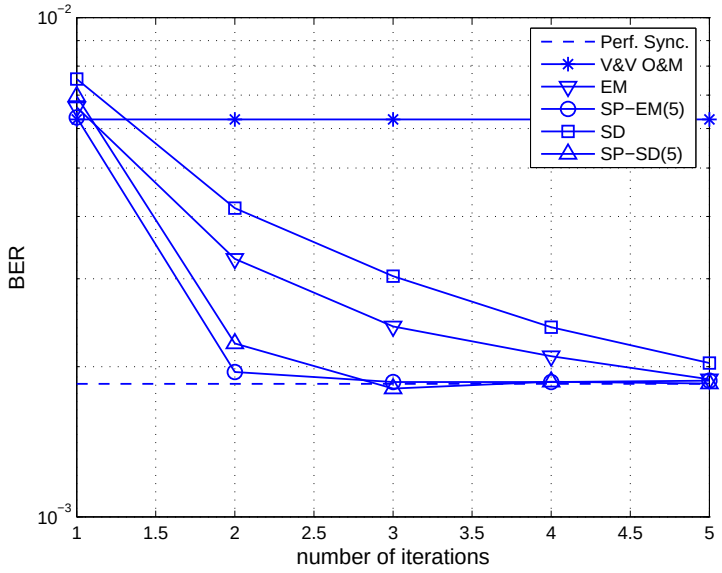

Figure 10.31 BER versus the number of iterations at $E_b/N_0 = 2.5$ dB.

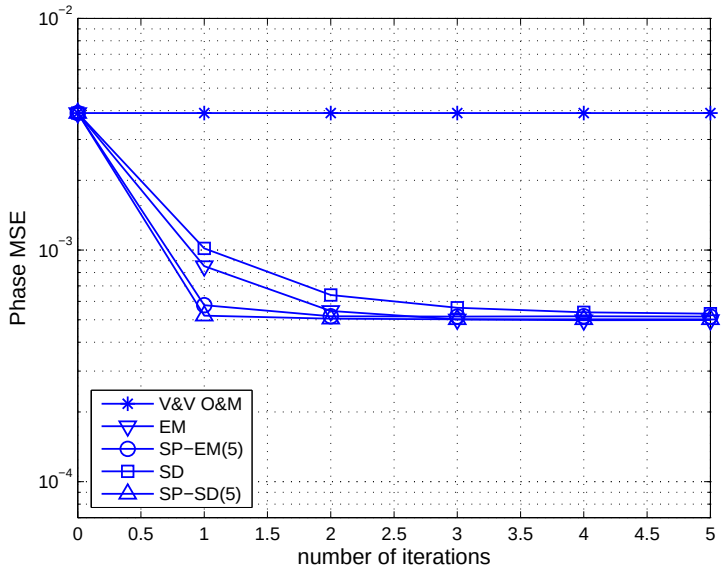

Figure 10.32 Carrier phase MSE versus the number of iterations at $E_b/N_0 = 2.5$ dB.

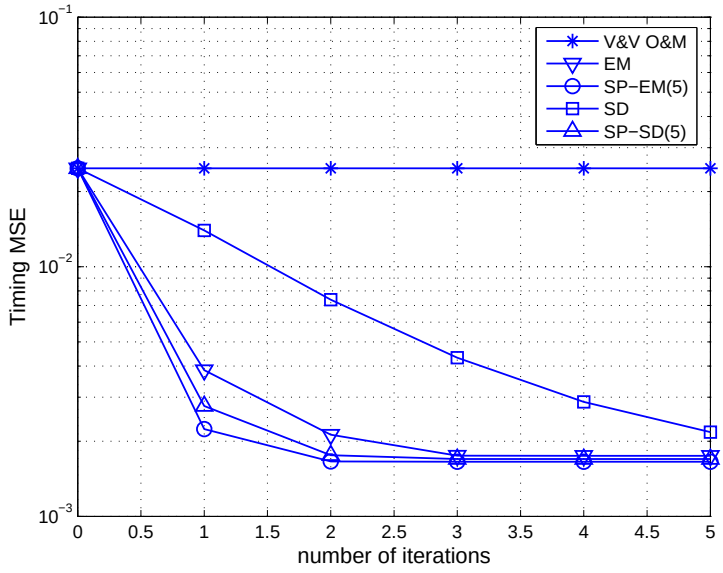

Figure 10.33 Timing MSE versus the number of iterations at $E_b/N_0 = 2.5$ dB.

that $|\tau - \hat{\tau}| \leq 0.5T$ and the phase estimate is such that $\|\theta - \hat{\theta}\| < \pi/4$.

We note that, unlike the conventional synchronizers, all the iterative SDD algorithms enable to recover the FER and the BER of a perfectly-synchronized system. The SDD synchronizers exhibit however a different behavior throughout the iterations. Fig. 10.30 and 10.31 represent the FER and the BER as a function of the number of iterations at $E_b/N_0 = 2.5$dB. We note that the different approaches have not the same speed of convergence. We first observe that the conventional synchronizers do not enable to improve the FER and the BER throughout the iterations since they are not iterative. A second observation is that only a few iterations are necessary to recover the performance of the perfectly-synchronized system: less than 5 iterations is usually sufficient. Note that the SP synchronizers exhibit a faster speed of convergence than the SD and the EM synchronizers.

The speed of convergence in terms of FER and BER may be related to the speed of convergence in terms of MSE. Fig. 10.32 and 10.33 respectively represents the MSE associated to the phase and timing estimation versus the number of iterations. The iteration labeled "0" is associated to the MSE before entering the decoder. In our simulation, the SDD synchronizers have been initialized by means of a V&V and an O&M synchronizer: the MSE at iteration 0 is therefore equal to the MSE achieved by these synchronizers. We note that the SP synchronizers have the fastest convergence. The EM synchronizer, although a little bit slower, converges quite rapidly. This fast converge is due to the system operating point which is such that BER$< 10^{-3}$. At such a low BER, it has been shown [12] that CRB$\simeq$MCRB. As discussed in section 10.2.1, this implies a low rate rate of convergence i.e. a fast speed of convergence. The SD synchronizer has the slowest speed of convergence. This is due to the difference of accuracy which can be achieved for the phase and the timing estimation. Indeed, the final MSE is about 5×10^{-4} for the phase against 2×10^{-3} for the timing. As a consequence, we can conclude that the Hessian matrix of the likelihood function has, in the average, an eigen value much smaller than the other one. As discussed in section 10.2.1, this implies a slow convergence in the direction of the timing. We observe this behavior in Fig. 10.33.

Turbo-coded Transmission

In this section we deal with the joint phase and timing synchronization of a turbo-coded transmission. We consider a turbo encoder which consists of

two rate-1/2 RSC encoders with polynomial generators $(37, 33)_8$, separated by an interleaver. The turbo encoder output is punctured so that the overall code rate is 1/2. Transmitted frames consist of 999 BPSK symbols. The roll-off factor is set to 0.1. The FER and the BER achieved by such a system when it is perfectly-synchronized are respectively illustrated in Fig. 10.34 and 10.35. 15 SP (or turbo) iterations are represented.

Fig. 10.36 and 10.37 represent the FER and the BER achieved at the 15^{th} SP iteration when the system is synchronized according to different approaches. The timing and the phase offset are changed for each new transmitted frame and are assumed to be uniformly distributed respectively on $[-0.5T, 0.5T]$ and on $[-\pi/2, \pi/2]$. We assume that the frame synchronization and the phase ambiguity problems are perfectly solved i.e. at each iteration the timing and the phase estimates are constrained to be such that $\|\tau - \hat{\tau}\| \leq 0.5T$ and $\|\theta - \hat{\theta}\| < \pi/2$. We consider the V&V and O&M, SP-EM, SP-SD and SP-NR synchronizers. The EM and the SD synchronizers are too complex to be implemented in practice and are not considered here. We note that the conventional synchronizers lead to degradation both in terms of FER and BER. On the contrary, the SDD synchronizers are able to recover the FER of the perfectly-synchronized system. In terms of BER, both the SP-EM and the SP-SD(1) synchronizers exhibit a gap with respect to the performance of the perfectly-synchronized system. Notice the very good performance of the SP-NR synchronizer: it recovers the BER of the perfectly-synchronized system by only performing one NR iteration per SP iteration.

Fig. 10.38 and 10.39 illustrate the FER and the BER versus the number of SP iterations at $E_b/N_0 = 2$ dB. Fig. 10.40 and 10.41 respectively represent the MSE associated to carrier-phase and timing estimation. In the latter figures, iteration 0 illustrates the synchronizer performance before entering the decoder. We note that:

- The system synchronized by means of the V&V and the O&M synchronizers can no longer improve the FER and the BER after a few iterations. This is due to the bad estimation quality of these synchronizers at such a low SNR: observe in particular the gap between the MSE achieved by the O&M and the SDD synchronizers. On the contrary, the systems synchronized by means of the proposed SDD synchronizers keep on improving the FER and the BER throughout the iterations. The speed of conver-

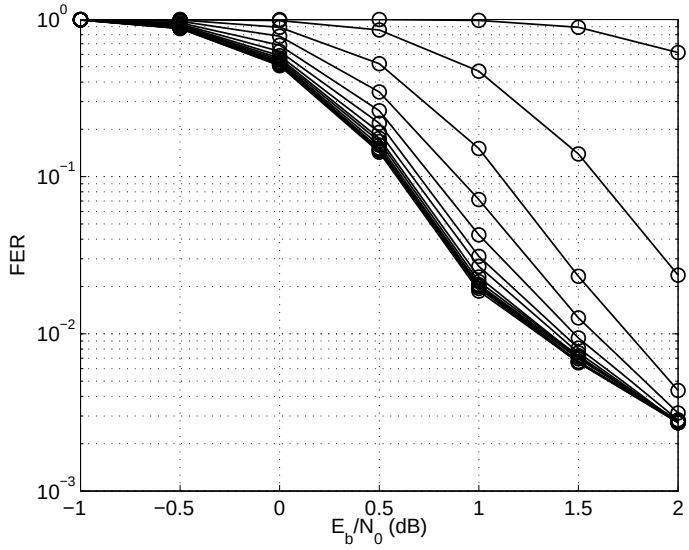

Figure 10.34 FER achieved at different (SP) iteration by a perfectly-synchronized turbo decoder. The turbo encoder consists in two punctured rate-1/2 RSC encoders with polynomial generators $(37, 33)_8$, separated by an interleaver.

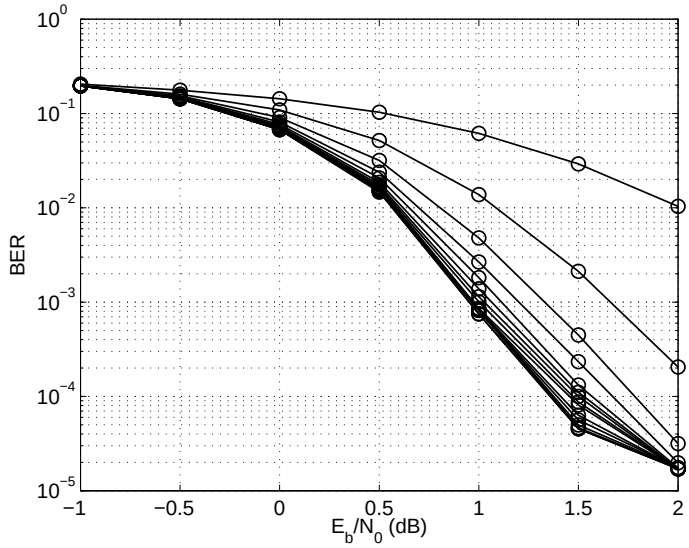

Figure 10.35 BER achieved at different (SP) iteration by a perfectly-synchronized turbo decoder. The turbo encoder consists in two punctured rate-1/2 RSC encoders with polynomial generators $(37, 33)_8$, separated by an interleaver.

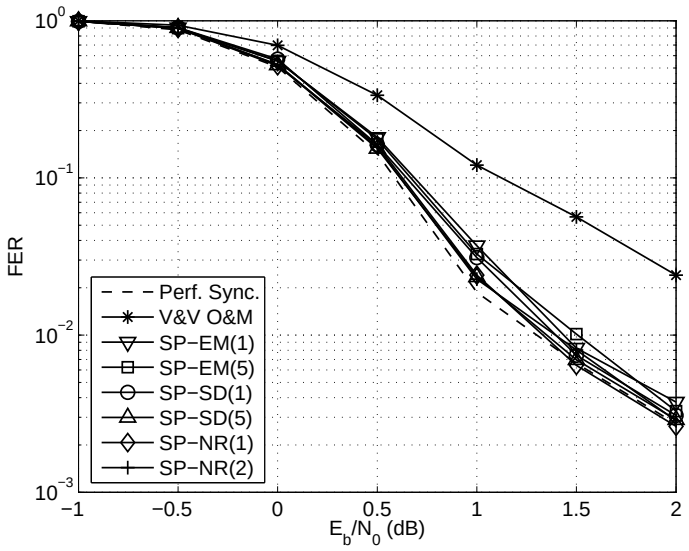

Figure 10.36 FER achieved by a turbo decoder at the 15th turbo (SP) iteration with different synchronization algorithms.

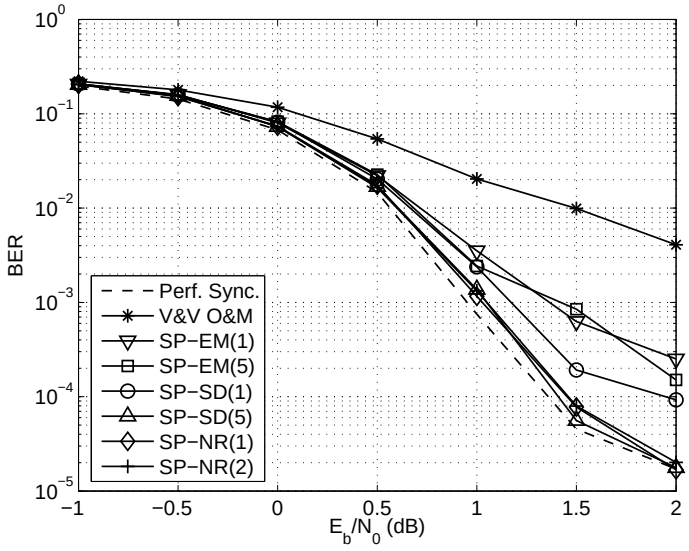

Figure 10.37 BER achieved by a turbo decoder at the 15th turbo (SP) iteration with different synchronization algorithms.

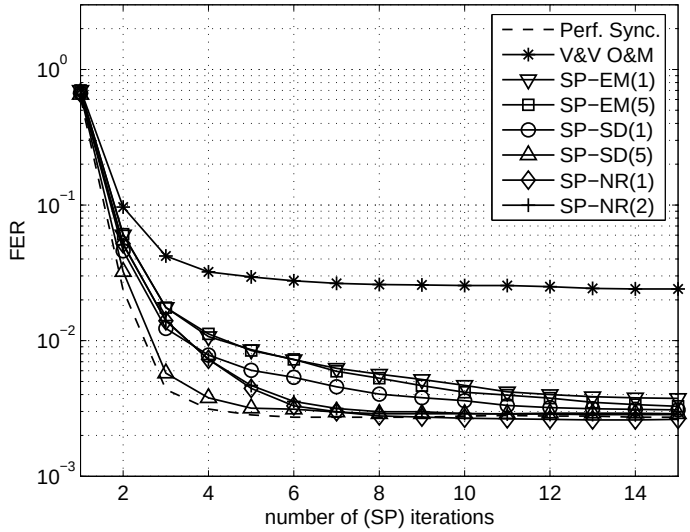

Figure 10.38 FER achieved by a turbo decoder at $E_b/N_0 = 2$ dB versus the number of SP iterations. The curves are plotted for different synchronization methods.

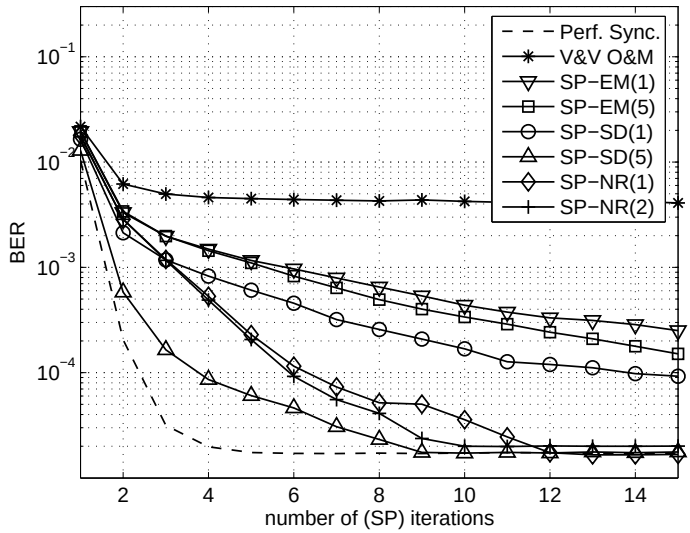

Figure 10.39 BER achieved by a turbo decoder at $E_b/N_0 = 2$ dB versus the number of SP iterations. The curves are plotted for different synchronization methods.

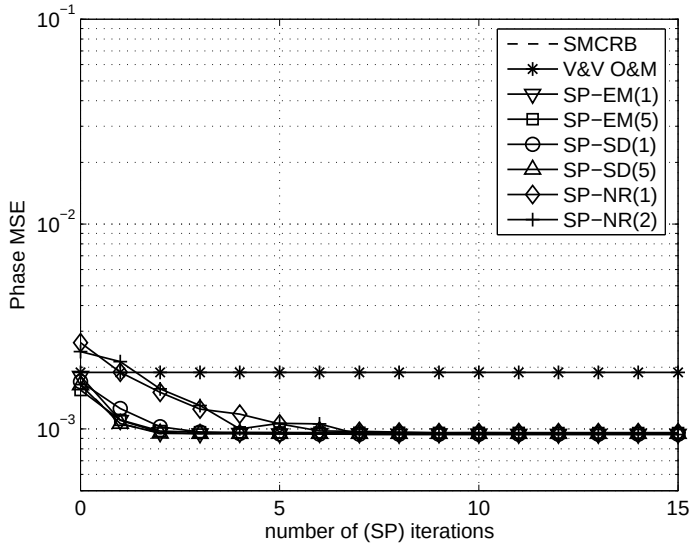

Figure 10.40 MSE associated to carrier-phase estimation versus the number of turbo (SP) iterations.

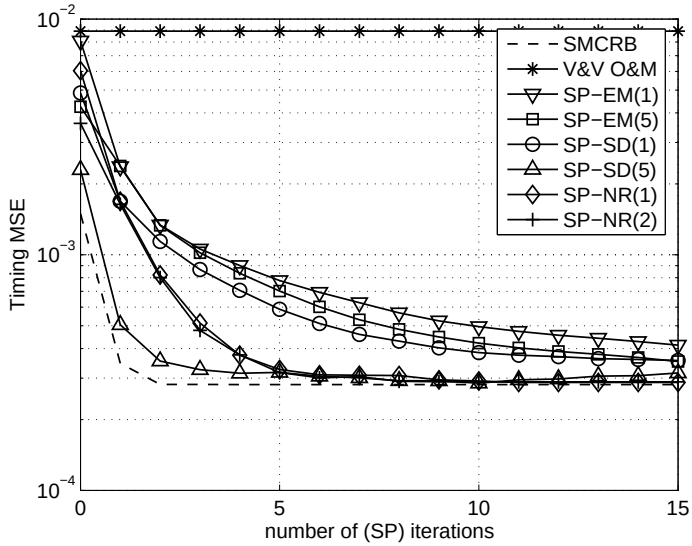

Figure 10.41 MSE associated to timing estimation versus the number of turbo (SP) iterations.

gence of the considered SDD methods is however quite different.

- The SP-EM algorithm has the slowest speed of convergence: after 15 SP iterations, it has still not recovered the BER of the perfectly synchronized system. This explains the gap observed in Fig. 10.37 between the BER achieved when using the SP-EM synchronizers and the BER of the perfectly-synchronized system. Note also the small improvement when increasing the number of EM iterations performed at each SP iteration. This behavior may be explained by the very low speed of convergence of the EM algorithm during the first SP iterations. Indeed, the less information about the transmitted symbol, the lower the speed of convergence of the EM algorithm. As a consequence, the SP-EM synchronizer fails in properly improving the estimation quality during the first iterations. Now, bad estimation quality during the first iteration may (as observed) slow down the convergence of the system. Indeed, the extrinsic information which is exchanged between the decoders at each SP iteration can keep a memory of bad initial estimates and therefore prevent the convergence of the system. We see therefore that the approach based on the EM algorithm keeps on decreasing very slowly the BER while exhibiting MSEs close to the MCRB.

- The same phenomena may be observed for the SP-SD(1) synchronizers: bad quality estimate during the first iterations slows down the convergence of the system. However, unlike the SP-EM synchronizer, we see that increasing the number of SD iterations per SP iteration improve the speed of convergence of the system. Notice in particular the FER achieved by using the SP-SD(5): it almost sticks the FER of the perfectly-synchronized system. This means that, using this synchronization technique, we do not have to perform additional decoding iterations to recover the FER of the perfectly-synchronized system.

- The SP-NR synchronizer converges fast. However, increasing the number of NR iterations per SP iteration does not seem to improve the speed of convergence. This behavior is actually due to the occasional divergence which may occur when using a NR algorithm: as mentioned above, the correction performed by the NR synchronizer may be quite inaccurate when the second derivatives of the likelihood function are small. This divergence may be observed in Fig. 10.40 where we see that the SP-NR does not stick to the SMCRB unlike the other synchronizers. This occasional divergence degrades the FER, BER and MSEs and leads

to a slow down of the mean speed of convergence.

Bit-interleaved Coded-Modulation Transmission

In this section, we apply different synchronizers to the problem of estimating the carrier phase offset[4] in a bit-interleaved coded-modulation (BICM) communication system. More particularly, we will try to emphasize the impact of the initialization procedure.

We consider a rate-1/2 non-systematic convolutional code with octal generators $(23, 35)_8$. The trellis of the code is terminated. The coded bits are interleaved by a random interleaver and mapped onto a 8-PSK constellation with a set-partioning mapping. At the receiver, we use a turbo demodulator to recover the transmitted bits. The latter device is made up with a soft-in-soft-out (SISO) demapper iteratively exchanging extrinsic information with a SISO decoder (see example 5.6 for more details). Fig. 10.42 and 10.43 illustrate the FER and the BER achieved at each iteration by such a communication system when the transmitted frames consist in 1000 8-PSK symbols and the synchronization parameters are known. 15 SP (or equivalently turbo) iterations are performed.

Fig. 10.44 and 10.45 respectively illustrate the FER and the BER achieved by the system with different synchronization algorithms, namely the V&V, the SP-EM, the SP-SD and the SP-NR. The EM and the SD synchronizers being too complex to be implemented in practice are not considered here. In this simulation, we have assumed that the phase-ambiguity resolution problem is perfectly solved so that we have $\|\hat{\theta} - \theta\| \leq 22.5^\circ$. The phase offset is uniformly distributed on $[-22.5, 22.5^\circ]$ and is changed for each new transmitted frame. The iterative synchronizers are initialized as follows: we evaluate the NDA log-likelihood function for θ equal to -17°, -6°, 6° and 17°; $\hat{\theta}^{(0)}$ is then set to the value which leads to the largest log-likelihood function. As in the previous example, we see that the conventional synchronization approach is unable to recover the FER and the BER of the perfectly-synchronized system. All the considered SSD synchronizers are able to reach the FER of the perfectly synchronized system. In terms of BER, we see that the SP-NR(1) synchronizer outperforms the SP-EM(5) and the SP-SD(5) synchronizers.

[4] All the other synchronization parameters are assumed to be perfectly known.

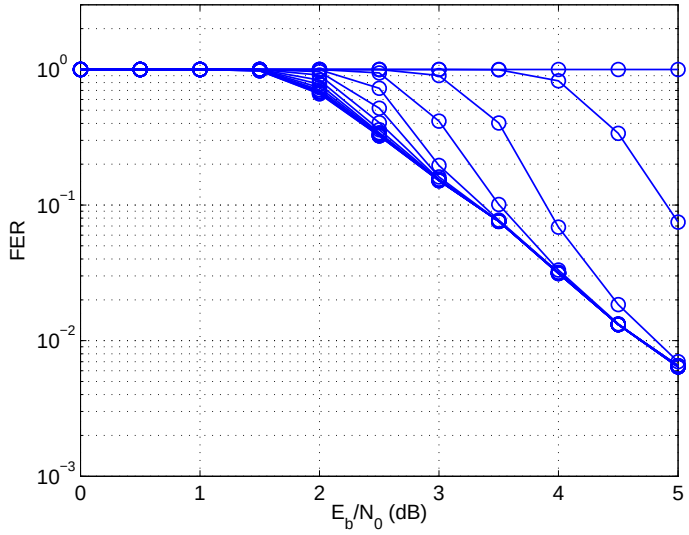

Figure 10.42 FER achieved by a perfectly-synchronized turbo demodulator for 8-PSK with set-partitioning mapping and a rate-1/2 length-5 convolutional code.

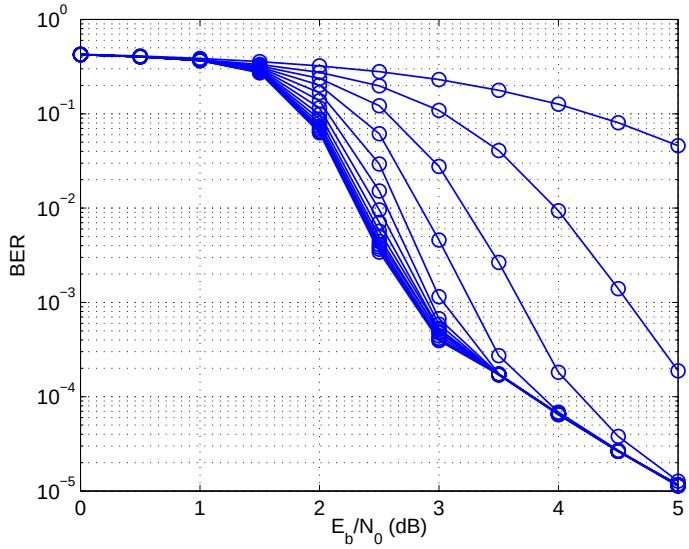

Figure 10.43 BER achieved by a perfectly-synchronized turbo demodulator for 8-PSK with set-partitioning mapping and a rate-1/2 length-5 convolutional code.

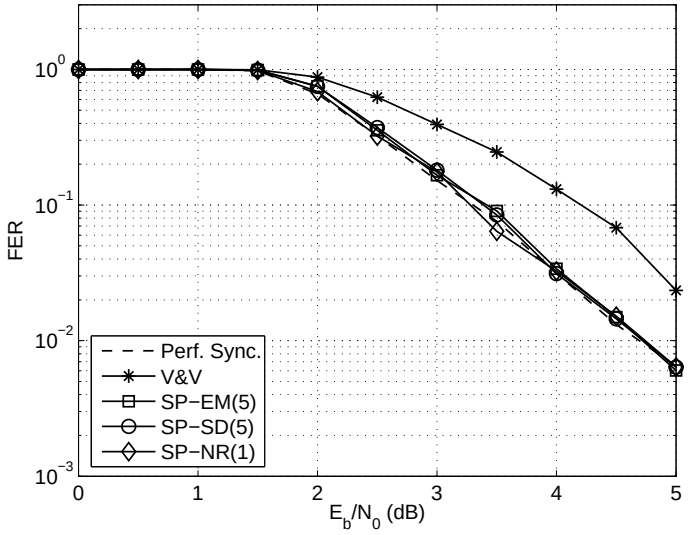

Figure 10.44 FER achieved by the turbo demodulator at the 15^{th} turbo (SP) iteration. The curves are plotted for different synchronization methods.

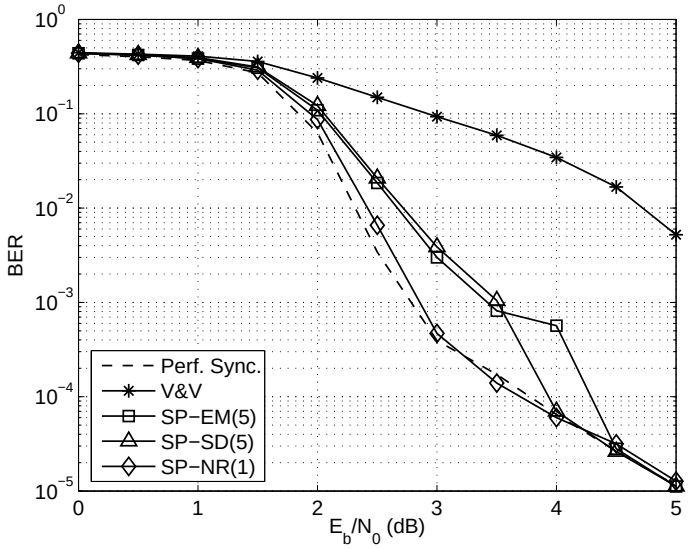

Figure 10.45 BER achieved by the turbo demodulator at the 15^{th} turbo (SP) iteration. The curves are plotted for different synchronization methods.

Fig. 10.46 and 10.47 show the FER and the BER versus the number of SP iterations at $E_b/N_0 = 5$ dB. We see that all the SDD synchronizers but the SP-EM(5) lead to the same performance in terms of FER. Regarding the BER, the synchronizer behavior differs. We may note:

- The SP-EM synchronizer converges slowly when too few EM iterations are performed per SP iteration. We see that the SP-EM(15) synchronizer decreases the BER very quickly whereas the SP-EM(5) synchronizer requires about 15 SP iterations to recover the BER of the perfectly-synchronized system. Note that the initialization procedure is such that the V&V synchronizer may be closer to the NDA ML estimate than the SDD synchronizer at iteration 0. Since the EM algorithm converges slowly during the first SP iterations, the estimation quality of this synchronizer may be quite bad so that the subsequent speed of convergence of the system is slowed down. This explains the difference between the SP-EM(5) and the SP-EM(15) synchronizers: the latter synchronizer enables to deliver better estimates during the first SP iterations and improve therefore the overall system convergence.
- The same kind of reasoning is valid regarding the behavior of the SP-SD synchronizers: too few SD iterations per SP iteration leads to bad estimation quality during the first iterations. As a consequence, we see in Fig. 10.47 that the SP-SD(1) is clearly outperformed by the SP-SD(5) and the SP-SD(15).
- The SP-NR synchronizer exhibits a very fast speed of convergence even if only 1 NR iteration is performed per SP iteration. Unlike in the previous example in which we considered the synchronization of a turbo decoder, we no longer observe a divergence in the behavior of the NR method. This is actually due to our initialization method. Indeed, on the one hand, although the V&V synchronizer leads in the average to better estimation quality, some of its estimates may be quite far from the maximum of the likelihood function. In such cases, the NR algorithm may diverge. On the other hand, the alternative initialization method which is used here ensures that the initial estimate is in an finite interval around the maximum of the likelihood function.

From the remarks above, we see that the initialization procedure is a crucial operation when using SDD synchronizers. In order to emphasize its impact, we have represented in Fig. 10.48 and Fig. 10.49 the FER and BER versus

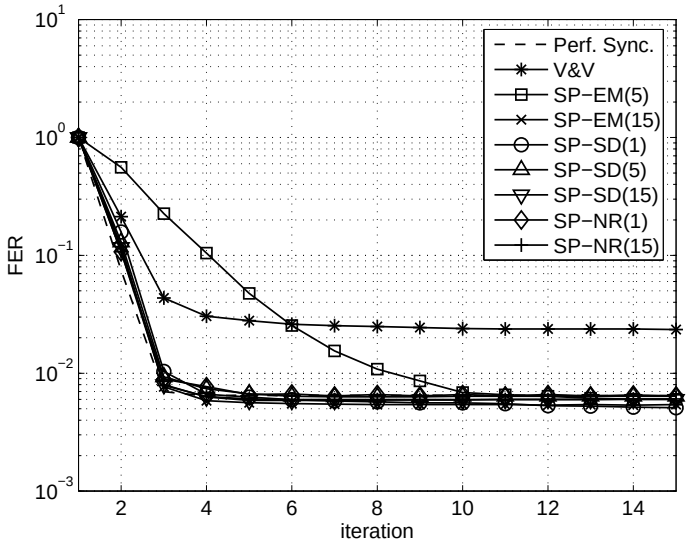

Figure 10.46 FER achieved by the turbo demodulator versus the number of turbo (SP) iterations at E_b/N_0 =5 dB. The curves are plotted for different synchronization methods.

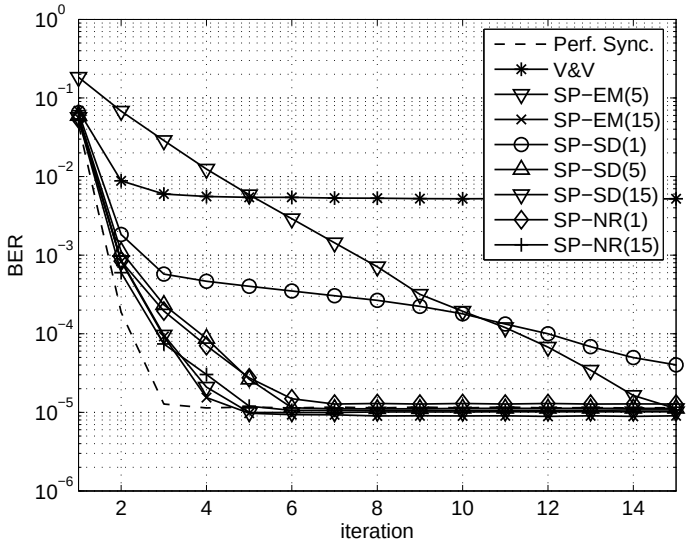

Figure 10.47 BER achieved by the turbo demodulator versus the number of turbo (SP) iterations at E_b/N_0 =5 dB. The curves are plotted for different synchronization methods.

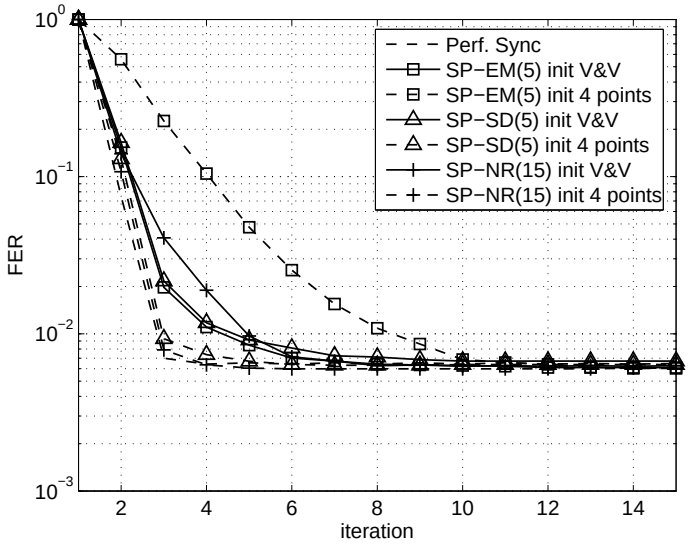

Figure 10.48 FER achieved by the turbo demodulator versus the number of turbo (SP) iterations at E_b/N_0 =5 dB. The curves are plotted for different synchronization methods and for different initialization procedures.

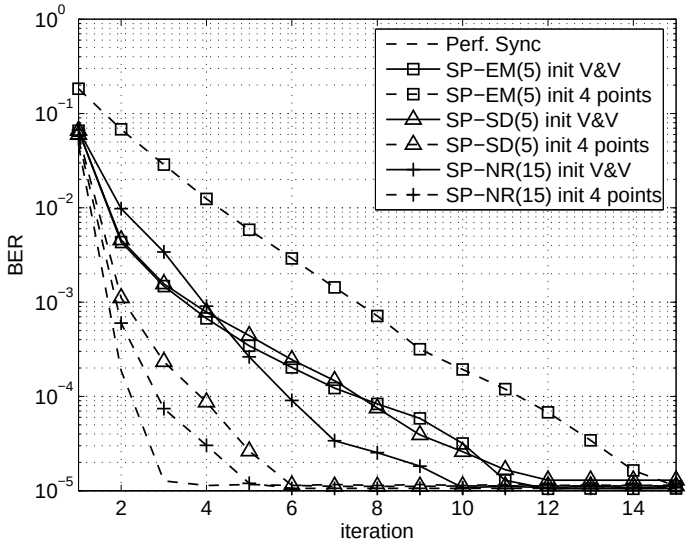

Figure 10.49 BER achieved by the turbo demodulator versus the number of turbo (SP) iterations at E_b/N_0 =5 dB. The curves are plotted for different synchronization methods and for different initialization procedures.

the number of SP iterations when the synchronizers are initialized according to 2 different ways. The curves labeled "init 4 points" are related to synchronizers initiated according to the method described above. The curves labeled "init V&V" correspond to synchronizers initialized via a V&V synchronizer. We see that the best initialization procedure depends on the considered SDD synchronizer. The SP-EM synchronizer has better performance when initialized with the V&V synchronizer. Indeed, the EM algorithm is robust but may have a very slow convergence. Therefore, the SP-EM synchronizer may take benefit from the better estimation quality of the V&V synchronizer. On the contrary, the SP-SD and SP-NR synchronizers exhibit better performance when initialized with the "4 points" method. Indeed, the SD and the NR algorithms are quite sensitive to the initial estimate: a bad initial value can lead to divergence. As a consequence, the "4 points" method which bounds the interval in which the initial estimate can be is better suited to the behavior of these algorithms.

10.3 Performance of Discrete-parameter Synchronizers

In this section we illustrate the performance of the discrete-parameter synchronizers proposed in chapter 8. In particular, we consider:

- *The FER-oriented MAP synchronizer:* this synchronizer is defined in (8.17). It consists in only maximizing a part of the likelihood function $\log p(\mathbf{b}|\mathbf{r})$. This synchronizer will be referred to as MAP_{FER} in the sequel.
- *The Maximum A Posteriori synchronizer:* the implementation of this synchronizer has been discussed in section 8.2.2. This synchronizer will be denoted by MAP in the sequel.
- *The FER-wise optimal synchronizer:* this synchronizer makes the decision about the synchronization parameters which minimizes the FER (see section 8.3.1). It will be denoted by FER_{opt} in the sequel.
- *The approximated FER-wise optimal synchronizer:* This approximated version of the FER_{opt} synchronizer is defined in (8.51) and is slightly less complex than the actual FER_{opt} synchronizer. This approach will be referred to as FER_{approx} in the sequel.

Note that the comparison of the performance of the FER-oriented MAP synchronizer with conventional DA and CA synchronizers has already been

treated in the current literature (see [84, 18, 85]). As a consequence, we will restrict our attention on the behavior of the synchronizers proposed in this work. In a first part we will illustrate the detection ability of CA synchronizers, i.e. the smallest probability of wrong detection that can be achieved. Then, we consider the synchronization of a convolutionally-coded transmission by means of the proposed CA synchronizers. Note that we only propose preliminary results in this work. A more complete study of the powerfulness of the proposed synchronization algorithms is let for future researches.

10.3.1 Detection Ability of CA Synchronizers

The performance of discrete-parameter estimators is usually measured in terms of the probability of making a wrong decision about the parameter value. For a given observation vector $\mathbf{r}$, the probability of making a wrong decision is given by

$$p(\text{wrong decision}|\mathbf{r}) = 1 - p(\mathbf{b} = \hat{\mathbf{b}}(\mathbf{r})|\mathbf{r}), \tag{10.6}$$

where $\hat{\mathbf{b}}(\mathbf{r})$ denotes the estimate of $\mathbf{b}$ computed from the observation vector $\mathbf{r}$. In the sequel, we will consider the minimum error probability which can be achieved by any CA synchronizer. The probability of wrong detection is minimized if we consider the MAP estimator (see chapter 3), i.e.

$$\begin{aligned} \hat{\mathbf{b}}(\mathbf{r}) &= \hat{\mathbf{b}}_{MAP}(\mathbf{r}), \\ &= \arg\max_{\tilde{\mathbf{b}}} p(\mathbf{b} = \tilde{\mathbf{b}}|\mathbf{r}). \end{aligned} \tag{10.7}$$

The smallest probability of wrong detection which can be achieved is therefore given by

$$p_{min}(\text{wrong decision}) = 1 - \int p(\mathbf{b} = \hat{\mathbf{b}}_{MAP}(\mathbf{r})|\mathbf{r})\, p(\mathbf{r})\, d\mathbf{r}. \tag{10.8}$$

In section 8.2.2, we show how to efficiently compute $p(\mathbf{b}|\mathbf{r})$. We can therefore compute $p_{min}(\text{wrong decision})$ by Monte-Carlo simulation, i.e. we compute $p(\mathbf{b}|\mathbf{r})$ for a large number of vectors $\mathbf{r}$ generated according to $p(\mathbf{r})$ and we compute the integral in (10.8) by an arithmetic average of the realization of $p(\mathbf{b}|\mathbf{r})$.

In Fig. 10.50 and Fig. 10.51, we illustrate $p_{min}(\text{wrong decision})$ in the case of carrier-phase ambiguity resolution. The results are represented versus the number of transmitted coded bits in Fig. 10.50 and versus the E_s/N_0-ratio in

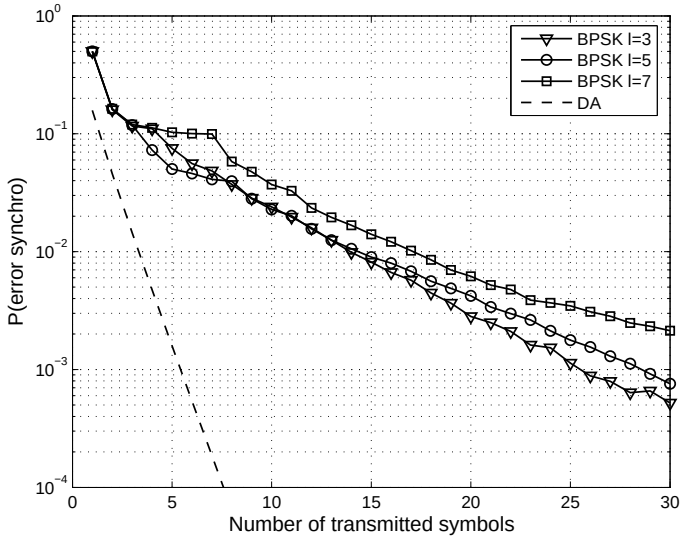

Figure 10.50 Smallest probability of wrong detection, i.e. $p_{min}(\text{wrong detection})$, versus the number of transmitted coded-bits. The curves are plotted for the DA case and for different constraint lengths.

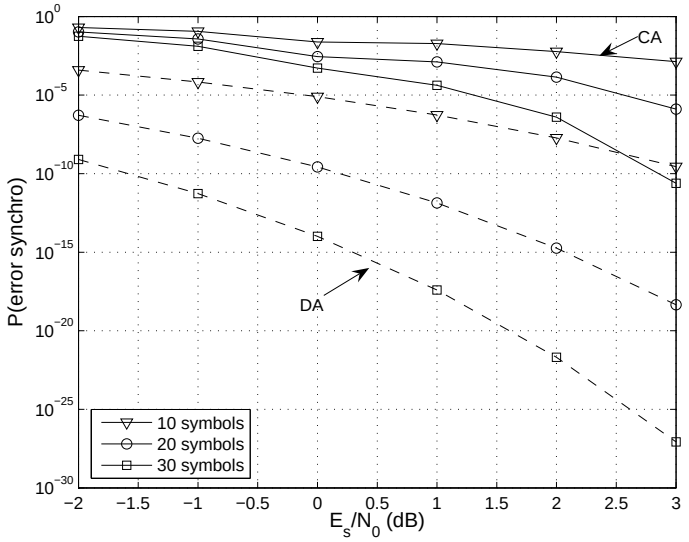

Figure 10.51 Smallest probability of wrong detection, i.e. $p_{min}(\text{wrong detection})$, versus the E_b/N_0-ratio. The curves are plotted for both the DA and the CA cases and for different frame length.

Fig. 10.51. The curves are represented for the DA case and non-systematic convolutional codes with constraint lengths respectively equal to 3, 5 and 7. The constellation alphabet is BPSK. In Fig. 10.50, the curves are plotted for $E_s/N_0 = 0$ dB. We see that $p_{min}(\text{wrong decision})$ keeps on decreasing as the number of transmitted bits increases. The slope of the CA curves is however much smaller than the one of the DA case. Therefore, as far as the considered case is concerned, increasing the number of transmitted pilots improves the synchronization performance much more rapidly than increasing the number of coded symbols taken into account in the CA synchronizer. Note however that pilot symbols lead to a waste power and spectral efficiency whereas coded symbols do not. Considering the performance achieved with the different codes, we see from Fig. 10.50 that increasing the constraint length does not improve the performance: the code with constraint length equal to 3 outperforms the codes with larger constraint lengths. The impact of the code structure on the synchronization performance should be addressed some further investigations. This topic is however beyond the scope of this work and will not be further addressed in the sequel. In Fig. 10.51, we have represented $p_{min}(\text{wrong decision})$ versus the E_s/N_0-ratio. The curves are plotted for different numbers of transmitted symbols. We see that, for a same amount of transmitted symbols, the slope of the CA and the DA cases is pretty similar. This means that increasing the transmitted power will (almost) affect in a similar way the performance of the DA and the CA synchronizers.

The previous results illustrate the average performance of the MAP CA synchronizer. We now consider the behavior of $p_{min}(\text{wrong decision}|\mathbf{r})$, i.e.

$$p_{min}(\text{wrong decision}|\mathbf{r}) = 1 - p(\mathbf{b} = \hat{\mathbf{b}}_{MAP}(\mathbf{r})|\mathbf{r}). \tag{10.9}$$

This probability is the minimum error probability given a particular realization of the observation vector $\mathbf{r}$. In the sequel, we will focus on how evolves this probability when we increase the number of transmitted symbols. Fig. 10.52 shows 4 different realizations of $p_{min}(\text{wrong decision}|\mathbf{r})$ as a function of the number of transmitted bits. We see that different realizations of $\mathbf{r}$ give rise to different behaviors of $p_{min}(\text{wrong decision}|\mathbf{r})$. In particular, for some realizations of $\mathbf{r}$ we can quite rapidly make a reliable decision on $\mathbf{b}$ whereas in other situations more symbols are required. If we want, for example, the probability of making a wrong decision to be as small as 10^{-3}, we see from Fig. 10.52 that we can make a decision with such a reliability by respectively taking 4, 5, 9 and 20 symbols into account. As a consequence, we see that

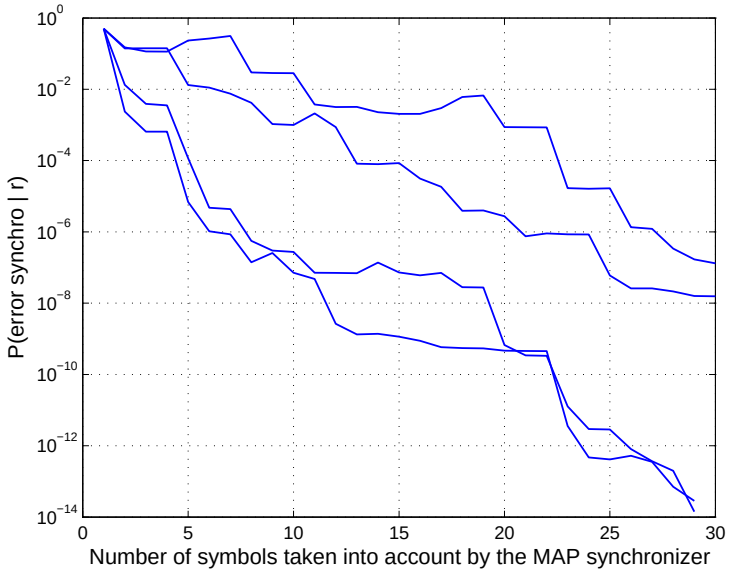

Figure 10.52 Probability of wrong detection of the MAP synchronizer for some realizations of the observation vector versus the number of symbols taken into account in the synchronizer operation.

processing the whole observation vector is not always necessary to get the required reliability on $\mathbf{b}$. Taking this into account, we see therefore that the method proposed in section 8.2.3 to only take into account a part of the transmitted symbols in the synchronization is relevant to decrease the complexity of the MAP synchronization algorithm.

10.3.2 Synchronization of a Convolutionally-coded Transmission

We focus on the phase-ambiguity resolution problem in a convolutionally-coded transmission. We consider 3 different rate-1/2 non-systematic convolutional codes with maximum free distance and constraint length respectively equal to 3, 5 and 7. We use a BPSK constellation alphabet. We apply the MAP_{FER}, the MAP, the FER_{opt} and the FER_{approx} to the problem of resolving the phase ambiguity. The continuous part of the carrier phase, i.e. ϵ_θ, is estimated by means of a Viterbi&Viterbi synchronizer.

Fig. 10.53 and 10.54 respectively represent the FER and the BER achieved by the communication system. We see from these figures that all the proposed

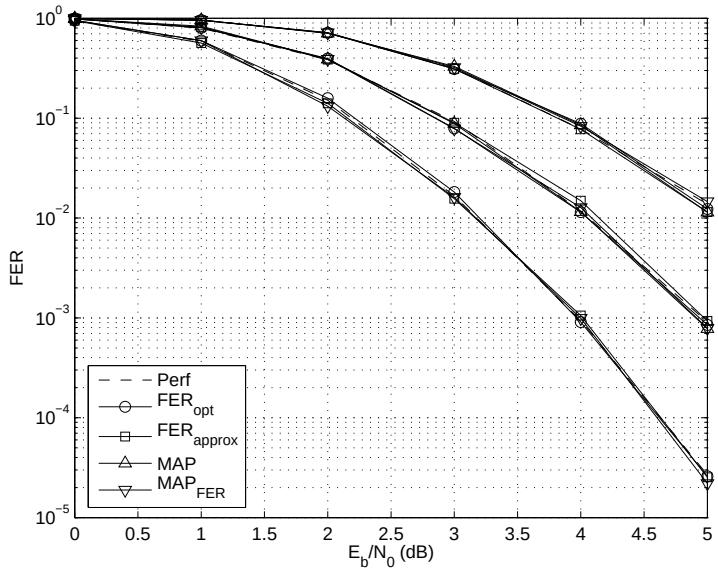

Figure 10.53 FER versus E_b/N_0-ratio for different synchronization algorithms.

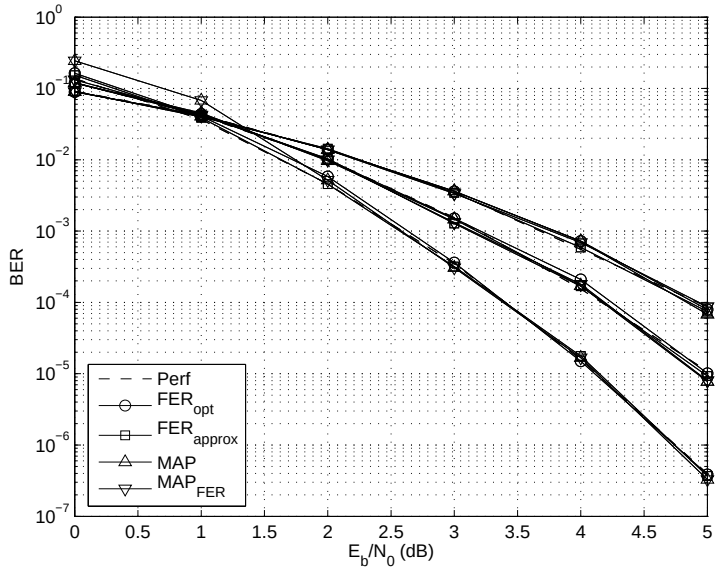

Figure 10.54 BER versus E_b/N_0-ratio for different synchronization algorithms.

algorithms enable to recover the performance of the perfectly synchronized system without the use of any pilot symbols. These approaches enable therefore to improve the power and/or the spectral efficiency of the system at the expense of an increase of the receiver complexity. The amount of complexity increase depends on the number of hypotheses to be tested. For instance, in the considered example there are two possible ambiguities, namely $k_\theta = 0$ and $k_\theta = 1$. We are therefore required to run the synchronizers for each of the hypotheses. The FER_{opt} and the FER_{approx} synchronizers both require two Viterbi decoding operations. Note that since we consider Viterbi decoding, only the latter two synchronization algorithms are well-suited to the decoding method. Indeed, unlike the FER_{opt} and the FER_{approx} synchronizers which may be easily implemented irrespective of the considered sequence decision rule, the MAP_{FER} and the MAP synchronizers require MAP decoding. If we had considered MAP decoding, the MAP synchronizer would have been the best solution in terms of complexity. Indeed, the test of each hypothesis only requires half the complexity of one decoding operation. Moreover, as shown previously, the decision about the synchronization parameters can be made without processing the whole frame.

Conclusions

11

During the last decade, capacity-approaching communication systems have turned into reality. On the one hand, powerful error-correcting codes, such as turbo code or LDPC codes, have been proposed. On the other hand, close-to-optimal iterative receiving structures have appeared. State-of-the-art communication systems are nowadays able to operate at low signal-to-noise ratios (SNRs) while exhibiting very low error rates.

In this context, synchronization has become a challenging operation: synchronizers are required to deliver accurate estimates of the synchronization parameters while operating at very low SNRs. As a direct consequence, previously-proposed synchronization algorithms, which worked well at medium-to-high SNRs, are no longer able to properly synchronize state-of-the-art receivers. This problem has called for novel efficient synchronization procedures.

This thesis has dealt with a new family of powerful synchronization algorithms: the code-aided (CA) synchronization algorithms. This kind of synchronizers attempts to take benefit from the structure of the error-correcting code used for the transmission to improve the estimation quality. Prior to this work, the problem of designing CA synchronizers had already been addressed in the literature. However, most of the proposed approaches were ad-hoc to some extent: they usually rested on some approximations or arbitrary assumptions which made quite difficult the prediction of their performance and/or behavior. Mathematical tools enabling both a systematic derivation of efficient CA synchronization algorithms and a way to predict and assess their performance were therefore still needed. This dissertation

have tackled this problem. The task has been divided into three main sub-problems: 1) the derivation of synchronization algorithms for continuous parameters (chapter 7); 2) the derivation of synchronization algorithms for discrete parameters (chapter 8); 3) The design of mathematical tools for assessing/predicting the performance of the proposed synchronizers (chapter 9).

The CA synchronization of continuous parameters has been placed in the context soft-decision-directed synchronization: we iteratively computed the CA maximum-likelihood (ML) estimate by taking benefit from soft symbol information available in the receiver. Three different frameworks have been proposed:

1. The first ML CA synchronization framework was based on the expectation-maximization (EM) algorithm. By nature, this algorithm ensures the convergence to a stationary point of the likelihood function. We discussed the exact implementation of the EM synchronizer and proposed an approximated low-complexity version of it. In a similar way, higher-dimensional synchronization problems have been placed in the context of the expectation conditional-maximization (ECM) algorithm.

2. A second ML CA synchronization framework was based on gradient algorithms. In particular, we considered the steepest-descent (SD) and the Newton-Raphson (NR) algorithms. These algorithms also ensure under fairly general conditions the convergence to a stationary point of the likelihood function. Like the EM synchronizer, the exact implementation has been discussed and an approximated low-complexity alternative proposed. In particular, we emphasized that the synchronizers based on the EM and the SD algorithms have the same computational complexity.

3. A last ML CA synchronization framework is based on the sum-product (SP) algorithm. The fixed points of these synchronizers have been studied: we have shown that the fixed points of the SP synchronizers are the stationary points of the likelihood function (as long as the ”synchronized” factor graph is cycle free). This kind of synchronizer has been shown to be easily implementable in state-of-the-art receivers which are based on a factor-graph representation.

   The SP synchronization framework has been finally combined with the

EM and gradient synchronization frameworks. Doing this, we showed that the low-complexity approximation of the EM (resp. gradient) synchronizer is a particular approximation of a more general framework based on both the SP and the EM (resp. gradient) algorithms. This observation has therefore allowed us to draw conclusions about the validity of the approximations made in the EM and gradient synchronizers.

Let us mention that several powerful soft-decision-directed synchronizers previously-proposed in an ad-hoc fashion in the literature have been shown to be particular instances of the synchronization frameworks developed in this thesis. As a conclusion, in the first part of this dissertation, we derived a set of synchronization frameworks enabling both to systematically derive CA ML iterative synchronizers and to understand/predict (up to a point) their performance.

In a second part of this dissertation, we have focused on the CA synchronization of discrete-valued parameters. Algorithms proposed in the literature were not optimal: they were either based on approximations of the MAP criterion or on the properties of some decoder metrics. In this work, we have proposed tractable synchronization algorithms which are optimal under different criteria:

1. We first considered CA MAP synchronization i.e. the optimal synchronization in terms of the minimization of the error probability on the decision made about the synchronization parameters. In a first part, we gave a (more) mathematical justification to an approximated MAP synchronizer recently-proposed in the literature. Then, we considered the resolution of the actual MAP estimation problem. We proposed an approach based on the SP algorithm to compute the MAP solution with a tractable complexity. The solution is either an approximation or is exact depending on whether the factor graph contains cycles or not. On the one hand, we showed that our algorithm to compute the MAP solution is well-suited to state-of-the-art receivers based on a factor-graph representation. On the other hand, we emphasized that the proposed approach has either the same or a lower complexity than other "MAP-based" synchronizers previously-proposed in the literature. Finally, we stressed that the proposed algorithm enables to easily compute intermediate MAP estimates on observation subspaces. This observation en-

ables to significantly decrease the complexity of the synchronizer by taking into account the reliability which is required on the decision made about the synchronization parameters.

2. In a second part, we considered optimal synchronization in terms of the minimization of the frame-error rate (FER). Although being the ultimate performance criterion in communication systems, the FER had never been considered before in the literature as a criterion for the design of synchronizers. We have emphasized that the FER-wise optimal synchronizer has a tractable complexity and, unlike approaches based on other criteria, may be easily implemented in any kind of receiver.

3. Finally, we considered the design of synchronizers optimal in the sense of the minimization of the bit-error rate (BER). On the one hand, we showed that this kind of synchronizer is tractable and well-suited to state-of-the-art receivers based on a factor-graph representation. On the other hand, we emphasized that the implementation of the BER-wise optimal receiver is also tractable and has actually the same complexity as a receiver based on a BER-wise optimal synchronization.

As a conclusion of the second part of this thesis, we have derived discrete-parameter synchronization algorithms which are optimal under different criteria. In particular, we have considered the minimization of the probability of wrong decision about the synchronization parameters, the minimization of the FER and the minimization of the BER. Although optimal, the proposed synchronization algorithms have been shown to have a complexity comparable or even lower than other existing synchronization methods.

A third goal of this thesis was to design mathematical tools to assess and predict the performance of soft-decision-directed synchronizers. In this work, we have derived lower bounds on the estimation mean square error (MSE) and on the speed of convergence, i.e.

1. We have considered two lower bounds on the estimation mean square error (MSE) achievable by any synchronizer handling some soft-information vector: the soft Cramer-Rao bound (SCRB) and the soft modified Cramer-Rao bound (SMCRB). The properties of these bounds have been analytically derived and their interest for assessing the performance of soft-decision-directed synchronizers shown by simulation

results.

2. We have also derived a lower bound on the speed of convergence of the EM synchronizer. This bound relates the speed of convergence of the synchronizer to the Cramer-Rao bound (CRB) and the modified CRB (MCRB) associated to the estimation problem. It therefore enables to easily predict the convergence behavior of the EM synchronizer in a number of practical cases.

In the last part of this dissertation, we have therefore derived analytical tools to assess and predict the performance of the SDD synchronizers proposed in chapter 7.

As a conclusion, this thesis has provided insightful answers to our initial questions (see Introduction). In particular, we show how to compute and combine symbol soft information to derive efficient SDD synchronizers. The properties of the proposed synchronizers are highlighted and discussed. In a last part, we derived some analytical tools providing a further insight into the behavior of CA synchronizers.

List of Publications

Journal Papers

- Cédric Herzet, Valéry Ramon and Luc Vandendorpe, "A Theoretical Framework for Iterative Synchronization based on the Sum-Product and the Expectation-Maximization Algorithms," accepted for publication in *IEEE Transactions on Signal Processing.*

- Cédric Herzet, Henk Wymeersch, Nele Noels, Marco Luise, Marc Moeneclaey and Luc Vandendorpe, "Turbo Synchronization or Iterative Code-aided ML Synchronization," accepted for publication in *The Transactions of the IEEE.*

- Cédric Herzet, Henk Wymeersch, Marc Moeneclaey and Luc Vandendorpe, "On Maximum-Likelihood Timing Estimation," accepted for publication in *IEEE Communications Letters.*

- Cédric Herzet, Henk Wymeersch, Frederik Simoens, Marc Moeneclaey and Luc Vandendorpe, "MAP-based Code-aided Hypothesis Testing in AWGN," submitted to *IEEE Communation Letters*.

- Valéry Ramon, Cédric Herzet and Luc Vandendorpe, "A Semi-analytical Prediction Method of Performance and Convergence Behavior of Multi-user Turbo-equalization/Demapping Scheme," submitted to *IEEE Transactions on Signal Processing*.

- Xavier Wautelet, Cédric Herzet, Antoine Dejonghe, Jérome Louveaux and Luc Vandendorpe, "Comparison of EM-based algorithms for MIMO channel estimation," submitted to *IEEE Transactions on Communications.*

Conference Papers

- Cédric Herzet, Xavier Wautelet, Valéry Ramon and Luc Vandendorpe, "FER-wise Optimal Code-Aided Hypothesis Testing," *ICC' 2006 - IEEE International Conference on Communications*, June 11-15, 2006, Istanbul, Turkey.
- Valéry Ramon, Aline Roumy, Cédric Herzet and Luc Vandendorpe, "Calculating the performance degradation of a MMSE/IC turbo-equalization scheme due to SNR estimation errors," *ICC' 2006 - IEEE International Conference on Communications*, June 11-15, 2006, Istanbul, Turkey.
- Cédric Herzet and Luc Vandendorpe, "Iterative Synchronization: EM Algorithm Versus Newton-Raphson Approach," *ICASSP'06 - IEEE Intl conf. on Acoustics, Speech and Signal Processing*, May 14-19, 2006, Toulouse, France.
- Valéry Ramon, Cédric Herzet, Xavier Wautelet and Luc Vandendorpe, "Calculating the performance of a soft-information-based best linear unbiased estimator of amplitude and carrier phase offset," *ICASSP'06 - IEEE Intl conf. on Acoustics, Speech and Signal Processing*, May 14-19, 2006, Toulouse, France.
- Cédric Herzet, Valéry Ramon and Luc Vandendorpe, "A Factor Graph Approach to Design Close-to-optimal Receivers in the Presence of a Timing Uncertainty," *EUSIPCO 2005 - European Signal Processing Conference*, September 4-8, 2005, Antalya, Turkey (Invited Paper).
- Cédric Herzet, Valéry Ramon and Luc Vandendorpe, "Turbo synchronization : A Combined Sum-Product and Expectation-Maximization Algorithm Approach," *SPAWC'05 - IEEE Workshop on Signal Processing Advances in Wireless Communications*, June 5-8, 2005, New-York, USA.
- Xavier Wautelet, Cédric Herzet and Luc Vandendorpe, "Cramer-Rao Bounds for Channel Estimation with Symbol a Priori Information," *SPAWC'05 - IEEE Workshop on Signal Processing Advances in Wireless Communications*, June 5-8, 2005, New-York, USA.
- Valéry Ramon, Cédric Herzet, Luc Vandendorpe and Marc Moeneclaey, "EM Algorithm-based Estimation of Carrier Phase, Amplitude and Noise," *ISSSTA'04 - IEEE International Symposium on Spread Spectrum Techniques and Applications*, September, 2004 , Sydney, Australia, pp. 550-554.

- Valéry Ramon, Cédric Herzet, Luc Vandendorpe and Marc Moeneclaey, "EM Algorithm-based Multiuser Synchronization in Turbo Receivers," *ICASSP'04 - IEEE Intl conf. on Acoustics, Speech and Signal Processing*, May 17-21, 2004, Montreal, Canada, vol. 4, pp. 849-852.
- Xavier Wautelet, Cédric Herzet, Antoine Dejonghe, Luc Vandendorpe, "MMSE-based and EM Iterative Channel Estimation Methods," *SCVT 2003 - IEEE 10th Symposium on Communications and Vehicular Technology in the Benelux*, November 13, 2003, Eindhoven, The Netherlands, SCVT032.pdf.
- Luc Vandendorpe, Antoine Dejonghe, Cédric Herzet, Xavier Wautelet and Marc Moeneclaey, "Soft-information-aided Synchronization: Recent Results," *SPSC'2003 - 8th Intl Workshop on Signal Processing for Space Communications*, September 24-26, 2003, Catania, Italy, pp. 195-202 (Invited paper).
- Cédric Herzet, Valéry Ramon, Luc Vandendorpe and Marc Moeneclaey, "An Iterative Soft-decision-directed Linear Timing Estimator," *SPAWC'03 - IEEE Workshop on Signal Processing Advances in Wireless Communications*, June, 2003, Roma, Italy.
- Nele Noels, Cédric Herzet, Antoine Dejonghe, Vincenzo Lottici, Luc Vandendorpe and Marc Moeneclaey, "Turbo Synchronization : an EM Algorithm Interpretation," *ICC'03 - IEEE Intl Conference on Communications*, May, 2003, Anchorage, USA, pp. 2933-2937.
- Cédric Herzet, Valéry Ramon, Luc Vandendorpe and Marc Moeneclaey, "EM algorithm-based Timing Synchronization in Turbo Receivers," *ICASSP'03 - ICASSP'03-IEEE International Conference on Acoustics, Speech and Signal Processing*, April, 2003, Hong Kong, China, vol. 4, pp. 612/615.
- Cédric Herzet, Valéry Ramon and Luc Vandendorpe, "Iterative Soft-decision-directed Timing Estimation for Turbo Receivers," *SCVT'02 - IEEE Symposium on Communications and Vehicular Technology in the Benelux*, October, 2002, Louvain-la-Neuve, Belgium, pp. 97-101.

COST Contributions

- Cédric Herzet, Valéry Ramon and Luc Vandendorpe, "Turbo-synchronization : an EM algorithm-based approach," *COST 273 - 6th*

Meeting of the Management Commitee, January 15-17, 2003, Barcelona, Spain, TD017.

- Cédric Herzet, Valéry Ramon and Luc Vandendorpe, "An Iterative Soft Decision Directed Linear Timing Estimator," *COST 273 - 7th Meeting of the Management Commitee*, September 23-26, 2003, Prague, Czech Rep., TD165.
- Cédric Herzet, Valéry Ramon and Luc Vandendorpe, "Message-passing Synchronization by Combining the SP and the EM Algorithms," *COST 273 - 12th Meeting of the Management Commitee*, January 19-21, 2005, Bologna, Italy, TD014.

Bibliography

[1] C.E. Shannon. "A Mathematical Theory of Communication". *Bell System Technical Journal*, 27:pp. 379–423, Jul. 1948.

[2] C. Berrou, A. Glavieux and P. Thitimajsima. "Near Shannon limit error-correcting coding and decoding". In *IEEE International Conference on Communications, ICC'*, pages pp. 1064–1070, Geneva, Switzerland, May 1993.

[3] S. ten Brink, J. Speidel, and J. C. Yan. Iterative demapping and decoding for multilevel modulation. In *IEEE Global Communications Conference, GLOBECOM'*, pages pp. 579–584, Sidney, Australia, November 1998.

[4] A. Glavieux, C. Laot and J. Labat. "Turbo equalization over a frequency selective channel". In *Int. Symp. on Turbo Codes & Rel. Topics, ISTC'*, pages pp. 96–102, Brest, France, September 1997.

[5] X. Wang and H.V. Poor. "Iterative (turbo) Soft Interference Cancellation and Decoding for Coded CDMA". *IEEE Trans. on Commun.*, 47(7):pp. 1046–1061, July 1999.

[6] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. "Factor graphs and the sum-product algorithm". *IEEE Trans. on Inform. Theory*, 47:pp. 498–519, February 2001.

[7] S.M. Aji, R.J. McEliece. "The Generalized Distributive Law". *IEEE Trans. on Inform. Theory*, 46(2):pp. 325–343, March 2000.

[8] H. Meyr, M. Moeneclaey, and S. Fetchel. *Digital Communication Receivers : Synchronization, Channel Estimation and Signal Processing*. Wiley Series in Telecommunications and Signal Processing, USA, 1998.

[9] U. Mengali and A.N. D'Andrea. *Synchronization Techniques for Digital Receivers*. Plenum Press, New York, USA, 1997.

[10] R. G. Gallager. "Low Density parity-check codes". *IRE Trans. Inform. Theory*, IT 8:pp. 21–29, Jan. 1962.

[11] D.J.C. MacKay. "Good Error-Correcting Codes Based on Very Sparse Matrices". *IEEE Trans. on Inform. Theory*, 45:pp. 399–431, March 1999.

[12] N. Noels, H. Steendam and M. Moeneclaey. "The Cramer-Rao Bound for Phase Estimation from Coded Linearly Modulated Signals". *IEEE Communication Letters*, 7(5):pp. 207–209, May 2003.

[13] N. Noels, H. Steendam and M. Moeneclaey. "The true Cramer-Rao Bound for Timing Recovery from a Bandlimited Linearly Modulated Waveform with Unknown Carrier Phase and Frequency". *IEEE Trans. on Commun.*, 52(3):pp. 473–483, March 2004.

[14] N. Noels, H. Steendam, M. Moeneclaey. "Carrier Phase Recovery in Turbo Receivers: Cramer-Rao Bound and Synchronizer performance". In *Int. Symp. on Turbo Codes & Rel. Topics, ISTC'*, Brest, France, Sep. 2003.

[15] V. Lottici and M. Luise. "Carrier phase recovery for turbo-coded linear modulations". In *IEEE International Conference on Communications, ICC'*, New-York, USA, April 2002.

[16] A. Burr and Zhang L. "A novel carrier phase recovery method for turbo-coded QPSK system". In *European Wireless, EW'*, Florence, Italy, February 2002.

[17] T.M. Cassaro and C.N. Georghiades. "Frame synchronization for coded systems over AWGN channel". *IEEE Trans. on Commun.*, 52(2):pp. 484–489, March 2004.

[18] H. Wymeersch and M. Moeneclaey. "Code-Aided frame synchronizers for AWGN channels". In *Int. Symp. on Turbo Codes & Rel. Topics, ISTC'*, Brest, France, September 2003.

[19] H. Wymeersch. "Software Radio Algorithms for Coded Transmission". PhD. thesis, Ghent University, Belgium, 2005.

[20] J. Dauwels. "On Graphical Models for Communications and Machine Learning: Algorithms, Bounds,and Analog Implementation". PhD. thesis, Swiss Federal Institute of Technology Zurich, 2005.

[21] J.G. Proakis. *Digital Communications-3rd ed.* McGraw-Hill International Editions, 1995.

[22] C. Berrou and A. Glavieux. "Near Optimum Error Correcting Coding and Decoding: Turbo Codes". In *IEEE Trans. on Commun.*, volume 44, pages pp. 1261–1271, October 1996.

[23] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, Cambridge, UK, 2005.

[24] A. Paulraj, R. Nabar, and D. Gore. *Introduction to Space-Time Wireless Communications*. Cambridge University Press, Cambridge, UK, 2003.

[25] S. Verdu. *Multiuser Detection*. Cambridge University Press, Cambridge, UK, 1998.

[26] S. Glisic and B. Vucetic. *Spread Spectrum Cdma Systems for Wireless Communications*. Artech House, MA, USA, 1997.

[27] J.A.C. Bingham. Multicarrier modulation for data transmission: An idea whose time has come. *IEEE Commun. Mag.*, 28(5):pp. 5–14, May 1990.

[28] R. Van Nee and R. Prasad. *OFDM for Wireless Multimedia Communications*. Artech House Publishers, 2000.

[29] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications, New York, USA, 1991.

[30] S. Benedetto and E. Biglieri. *Principles of Digital Transmission With Wireless Applications*. Kluwer Academic/ Plenum Publishers, New York, USA, 1999.

[31] X. Jaspar and L. Vandendorpe. "Design and Performance Analysis of Joint Source-Channel Turbo Schemes with Variable Length Codes". In *Proc. IEEE ICC'2005*, Seoul, Korea, May 2005.

[32] R. Bauer and J. Hagenauer. "Iterative Source/Channel Decoding Using Reversible Variable Length Codes". In *Proc. IEEE DCC'2005*, Snowbird, USA, March 2000.

[33] H. Cramer. "A contribution to the theory of statistical estimation". *Skand. Akt. Tidskr*, 29:pp. 85–94, 1945.

[34] C.R. Rao. Information and accuracy attainable in the estimation of statistical parameters. *Bull. Calcutta Math. Soc.*, 37:pp. 81–91, 1945.

[35] J.M. Mendel. *Lessons in Estimation Theory for Signal Processing Communications and Control*. Prentice Hall Signal Processing Series, Englewood Cliffs, NJ, 1995.

[36] R.L. Kashyap and A.R. Rao. *Dynamic Stochastic Models from Empirical Data*. Academic Press, New York, USA, 1976.

[37] T.S. Ferguson. *Mathematical Statistics: A Decision Theroetic Approach*. Academic Press, New York, USA, 1967.

[38] R.D. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience, New York, USA, 1973.

[39] E.W. Barankin. "Locally best unbiased estimates". *Ann. Math. Statist.*, 20:pp. 477–501, 1949.

[40] A. Bhattacharyya. "On some analogues of the amount of information and their use in statistical estimation". *Shankya*, 8(1):pp. 1–14, 201–211, 211–218, 1946.

[41] H.L. Van Trees. *Detection, Estimation and Modulation Theory: Part 1*. Wiley, New York, USA, 1966.

[42] A. D'Andrea, U. Mengali and R. Reggiannini. "The Modified Cramer-Rao Bound and Its Application to Synchronization Problems". *IEEE Trans. Comm.*, 42(2/3/4):pp. 1391–1399, February/March/April 1994.

[43] R. Ravikanth and S.P. Meyn. "Bounds on achievable performance in the identification and adaptive control of time-varying systems". *IEEE Trans. Automatic Control*, 44:pp. 670–682, 1999.

[44] M. Moeneclaey. "On the true and the modified Cramer-Rao bounds for the estimation of a scalar parameter in the presence of nuisance parameters". *IEEE Trans. on Commun.*, 46(11):pp. 1536–1544, November 1998.

[45] G.D. Forney. "The Viterbi Algorithm". *Proc. of the IEEE*, 61(3):pp. 268–278, March 1973.

[46] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate". *IEEE Trans. on Inform. Theory*, 20:pp. 284–287, March 1974.

[47] D.P. Bertsekas. *Nonlinear Programming*. Athena Scientific, USA, 2003.

[48] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum-likelihood from incomplete data via the EM algorithm". *J. Roy. Stat. Soc.*, 39(1):pp. 1–38, January 1977.

[49] W.I. Zangwill. *Nonlinear Programming: A Unified Approach*. Prentice Hall, Englewood Cliffs, New Jersey, USA, 1969.

[50] C. F. J. Wu. "On the convergence properties of the EM algorithm". *Ann. Statistics*, 11(1):pp. 95–103, 1983.

[51] G.J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley Series in Probability and Statistics, USA, 1997.

[52] X.L. Meng and D.B. Rubin. "Maximum Likelihood Estimation via the ECM algorithm: a general framework". *Biometrika*, 80:pp. 267–268, 1993.

[53] J. A. Fessler and A. O. Hero. "Space-alternating generalized expectation-maximization algorithm". *IEEE Trans. Signal Processing*, 42(10):pp. 2664–2677, Oct. 1994.

[54] X.L. Meng and D. van Dyk. "The EM algorithm - An old folk song sung to a fast new tune". Technical Report No. 408, Department of Statistics, University of Chicago, USA, 1995.

[55] M. Oerder and H. Meyr. "Digital filter and square timing recovery". *IEEE Trans. Comm.*, 36:605–611, May 1988.

[56] A. J. Viterbi and A. M. Viterbi. "Nonlinear estimation of PSK-Modulated Carrier Phase with Application to Burst Digital Transmission". *IEEE Trans. on Inform. Theory*, 29:pp. 543–551, Jul. 1983.

[57] M. Morelli, A. D'Andrea and U. Mengali. "Feedforward ML-based timing estimation with PSK signals". *IEEE Trans. Comm.*, 1(3):pp. 80–82, May 1997.

[58] L. Zhang and A. Burr. "Joint Decoding and Phase Recovery for Turbo-coded BPSK". In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'*, Salt Lake City, USA, May. 2001.

[59] A.W. Eckford. "Channel Estimation in Block Fading Channels Using Factor Graph EM Algorithm". In *Proc. 22nd Biennal Symposium on Communications*, Kingston, Canada, 2004.

[60] J. Dauwels and S.Korl and H.-A. Loeliger. "Expectation-Maximization as Message Passing". In *IEEE Int. Symp. on Information Theory, ISIT'*, Adelaide, Australia, September 2005.

[61] N. Noels, C. Herzet, A. Dejonghe, V. Lottici, H. Steendam, M. Moeneclaey, M Luise and L. Vandendorpe. "Turbo-synchronization: an EM algorithm approach". In *Proc. IEEE ICC*, Anchorage, May 2003.

[62] C. Herzet, V. Ramon, L. Vandendorpe and M. Moeneclaey. "EM algorithm-based timing synchronization in turbo receivers". In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'*, Hong Kong, April 2003.

[63] V. Ramon, C. Herzet, L. Vandendorpe and M. Moeneclaey. "EM algorithm-based multiuser synchronization in turbo receivers". In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'*, Montreal, April 2004.

[64] M. Guenach, H. Wymeersch, and M. Moeneclaey. "Joint Estimation of Path Delay and Complex Gain for Coded Systems Using the EM Algorithm". In *Proc. International Zurich Seminar, IZS'*, Zurich, Switserland, February 2004.

[65] M. Guenach, F. Simoens, H. Wymeersch, and M. Moeneclaey. "Code-aided Frame Joint Channel and Frequency Estimation for ST-BISM multi-user DS-CDMA system". In *EURASIP European Signal Processing Conference, EUSIPCO'*, Antalya, Turkey, September 2005.

[66] A. Papoulis. *Probability, Random Variables, and Stochastic Processes-3rd ed.* McGraw-Hill International Editions, 1991.

[67] J. Dauwels and H.-A. Loeliger. "Phase Estimation by Message Passing". In *IEEE International Conference on Communications, ICC'*, Paris, France, June 2004.

[68] A. P. Worthen, W. E. Stark. "Unified Design of Iterative Receivers Using Factor Graphs". *IEEE Trans. on Inform. Theory*, 47(2):pp. 843–849, 2001.

[69] G. Colavolpe and A. Barbieri and G. Caire. "Algorithms for iterative decoding in the presence of strong phase noise". *IEEE J. Select. Areas Commun.*, 23:pp. 1748–1757, Sep. 2005.

[70] J. L. Massey. "Optimum frame synchronization". *IEEE Trans. on Commun.*, 20:pp. 115–119, Apr. 1972.

[71] J. L. Lui, and H. H. Tan. "Frame Synchronization for Gaussian Channel". *IEEE Trans. on Commun.*, 35(8):pp. 818–829, August 1987.

[72] A. Sandri U. Mengali and A. Spalvieri. "Phase Ambiguity Resolution in Treillis-Coded Modulations". *IEEE Trans. on Commun.*, 38(12):pp. 2087–2089, Dec. 1990.

[73] G. Lorden, R.J. McEliece, and L. Swanson. "Node Synchronization for the Viterbi Decoder". *IEEE Trans. on Commun.*, 32:pp. 524–531, May 1984.

[74] B. Mielczarek and A. Svensson. "Timing error recovery in turbo coded systems on AWGN channels". *IEEE Trans. Comm.*, 50(10):pp. 1584–1592, Oct. 2002.

[75] U. Mengali, R. Pellizzoni and A. Spalvieri. "Soft-Decision-Based Node Synchronization for Viterbi Decoders". *IEEE Trans. on Commun.*, 43(9):pp. 2532–2539, Sept. 1995.

[76] J. Dauwels, H. Wymeersch, H.-A. Loeliger and M. Moeneclaey. "Phase Estimation and Phase Ambiguity Resolution by Message Passing". In *Proc. 11th International Conference on Telecommunications and Networking*, Fortaleza, Brazil, August 2004.

[77] P. Robertson. "A Generalized Frame Synchronizer". In *IEEE Global Communications Conference, GLOBECOM'*, pages pp. 365–369, December 1992.

[78] P. Robertson. "Improving frame synchronization when using convolutional codes". In *IEEE Global Communications Conference, GLOBECOM'*, pages pp. 1606–1611, December 1993.

[79] H. Wymeersch and H. Steendam and H. Bruneel and M. Moeneclaey. "Code-Aided Frame Synchronization and Phase Ambiguity Resolution". to appear inIEEE Trans. Signal Processing.

[80] W.G. Cowley. "Phase and frequency estimation for PSK packets: Bounds and algorithms". *IEEE Trans. on Commun.*, 44:pp. 26–28, Jan. 1996.

[81] F. Rice, B. Cowley, B. Moran and M. Rice. "Cramer-Rao lower bounds for QAM phase and frequency estimation". *IEEE Trans. on Commun.*, 49:pp. 1582–1591, Sep. 2001.

[82] N. Noels, H. Steendam and M. Moeneclaey. "Carrier and Clock recovery in (turbo) coded systems: Cramer-Rao Bound and Synchronizer Performance". accepted for publication in *EURASIP Journal on Applied Signal processing*, available at http://telin.rug.ac.be/ nnoels/articles.html.

[83] S. ten Brink. "Convergence behavior of iteratively decoded parallel concatenated codes". *IEEE Trans. on Commun.*, 49:pp. 1727–1737, October 2001.

[84] H. Wymeersch and M. Moeneclaey. "ML frame synchronization for turbo and LDPC codes". In *7th International Symposium of Digital Signal Processing and Communication Systems (DSPCS'03)*, Coolangatta, Australia, December 2003.

[85] H. Wymeersch and M. Moeneclaey. "Iterative Code-Aided ML Phase Estimation and Phase Ambiguity Resolution". *EURASIP Journal on Applied Signal Processing*, 2005(6), 2005.