INVITED PAPER

Code-Aided Turbo Synchronization

Techniques that combine both data detection and synchronization in an iterative way can improve the overall accuracy and performance of the system.

By Cédric Herzet, Nele Noels, Vincenzo Lottici, Henk Wymeersch, Marco Luise, *Senior Member IEEE*, Marc Moeneclaey, *Fellow IEEE*, and Luc Vandendorpe, *Fellow IEEE*

ABSTRACT | The introduction of turbo and low-density parity-check (LDPC) codes with iterative decoding that almost attain Shannon capacity challenges the synchronization subsystems of a data modem. Fast and accurate signal synchronization has to be performed at a much lower value of signal-to-noise ratio (SNR) than in previous less efficiently coded systems. The solution to this issue is developing specific synchronization techniques that take advantage of the presence of the channel code and of the iterative nature of decoding: the so-called *turbo-synchronization* algorithms. The aim of this paper within this special issue devoted to the turbo principle is twofold: on the one hand, it shows how the many turbo-synchronization algorithms that have already appeared in the literature can be cast into a simple and rigorous theoretical framework. On the other hand, it shows the application of such techniques in a few simple cases, and evaluates improvement that can be obtained from them, especially in the low-SNR regime.

Manuscript received November 3, 2006; revised February 10, 2007. This work was supported in part by the Interuniversity Attraction Poles Program P5/11—Belgian State—Federal Office for Scientific, Technical and Cultural Affairs and in part by the EU FP6 Network of Excellence NEWCOM.

C. Herzet is with the Communications Laboratory, Université Catholique de Louvain, B-1348 Louvain-la-Neuve, Belgium, and also with the Wireless Foundations—Electrical Engineering and Computer Science Department, University of California Berkeley, Berkeley, CA 94720 USA (e-mail: herzet@eecs.berkeley.edu).

N. Noels and **M. Moeneclaey** are with DIGCOM Research Group, TELIN Department, Ghent University, B-9000 Ghent, Belgium (e-mail: nnoels@telin.UGent.be; mm@telin.UGent.be).

V. Lottici is with the Dipartimento di Ingegneria dell'Informazione, University of Pisa, I-56122 Pisa, Italy (e-mail: vincenzo.lottici@iet.unipi.it).

H. Wymeersch is with DIGCOM Research Group, TELIN Department, Ghent University, B-9000 Ghent, Belgium, and also with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: hwymeers@mit.edu; http://www.mit.edu/~hwymeers/).

M. Luise is with the Dipartimento di Ingegneria dell'Informazione, University of Pisa, I-56122 Pisa, Italy (e-mail: m.luise@iet.unipi.it; http://www.iet.unipi.it/~luise).

L. Vandendorpe is with the Communications and Remote Sensing Laboratory, Université Catholique de Louvain, 1348 Louvain-la-Neuve, Belgium (e-mail: Vandendorpe@tele.ucl.ac.be).

Digital Object Identifier: 10.1109/JPROC.2007.896518

KEYWORDS | Iterative methods; maximum likelihood estimation; synchronization

I. INTRODUCTION

Synchronization, from the Greek *synchronos* [i.e., *syn* (together) + *chronos* (time)] denotes the function of making two systems or two signals running exactly together at the same pace. Specifically, the synchronization subsystems of a modem have the purpose of achieving the correct alignment of the incoming waveform with certain locally generated references [1], [2]. For instance, in bandpass transmission a coherent receiver needs *carrier synchronization*, which means that the sinusoid for bandpass-to-baseband conversion must be locked in phase and frequency to the incoming carrier. Good signal synchronization often turns out to be the key to developing a good modem with fast signal acquisition and smooth steady-state operation. How is this related to the main subject of this special issue, namely, the turbo principle?

The introduction of turbo codes in the mid-1990s marked the beginning of a lot of activity in the field of research, development, and standardization addressing the performance analysis, the design, and the application of iterative decoding in digital communications [3]–[5]. The impressive performance of coded signals when applied to (wireless) communication terminals, however, implicitly assumes ideal synchronization of the received signal. So the adoption of powerful channel codes such as turbo and low-density parity-check (LDPC) codes [6]–[8] engenders a main issue for the synchronization subsystems of a modem: how to derive accurate references from the received signal at those (extremely low) SNRs typical of such codes? Furthermore, how to do this with a reasonably

short acquisition time, possibly using a short preamble of known (pilot) data, or even no preamble at all?

Earlier attempts of signal synchronization in the low-SNR regime focused either on the so-called *data-aided (DA)* or *non-data-aided (NDA) synchronization mode* [1], [2]. On the one hand, DA parameter estimation techniques rely on the presence of pilot symbols in the data frame and may lead to unacceptable losses in terms of power and spectral efficiency. On the other hand, NDA synchronization algorithms drop some statistical information about the transmitted data and may lead to very poor results at low SNR. As a consequence, it was soon recognized [9], [10] that the only way to get good performance both in terms of acquisition time and steady-state accuracy (the so-called *jitter variance*) is taking advantage of the "coding gain" not only for data detection, but for synchronization as well. This led us to the notion of *code-aided* (CA) synchronization, as explicitly mentioned in the title of this paper: explicitly using the channel code structure and properties to perform good non-pilot-aided (NPA) synchronization.

Although NPA CA synchronization (hereinafter CA for short) may appear as a natural solution to improve synchronization functions, its implementation is at times critical and has led to numerous contributions in the technical literature, see, e.g., [11] and [12]. Among these algorithms, two main approaches can be distinguished.

A first approach consists in modifying the detection/decoding device in order to embed parameter estimation. In [11] and [13], for example, combined iterative decoding and estimation is performed by modifying the decoder using a sort of per-survivor estimation technique. In [14], using a simplified error phase model, the authors propose to output soft-values of the decoders giving an indication of the sign of the phase error. In [15], the authors propose a method with polynomial complexity to generate soft symbol-information taking into account the synchronization-parameter uncertainty.

A second category of algorithms proposed in the literature is based on the estimation of the synchronization parameters from some information outputs by the decoder. Early attempts of CA synchronization based on this approach used (possibly iteratively) hard symbol decisions computed by the decoder, see, e.g, [16] and [17]. The "intertwine" of channel decoding and synchronization was further refined in the context of turbo receivers, see, e.g., [18] and [19]. The idea was as follows: instead of exploiting hard decisions on the transmitted symbols (which implies a loss of information about the decision reliability), the authors proposed to feed the synchronizer with the so-called *soft* symbol decisions carrying both the symbol decision and its reliability, and to refine the estimate in an iterative fashion, as long as the reliability of the soft symbols improves due to turbo decoding. This approach was soon referred to as *turbo synchronization* in agreement with the turbo principle. Earlier approaches for turbo synchronization aimed at plugging the soft-outputs of an iterative decoder into the structure of a conventional synchronizer to gain back reliability with respect to standard symbols hard-decisions used in a decision-directed algorithm [18], [19]. Note that similar ideas simultaneously appeared for the closely related problem of channel estimation, see, e.g., [20] and [21]. Although somewhat *ad hoc*, these approaches were shown to lead to performance improvement with respect to conventional NDA synchronization methods. What was however missing was a general framework to justify the architecture of such synchronizers and to suggest the development of new algorithms for new synchronization and estimation problems (signal amplitude, carrier frequency, channel frequency response in multicarrier systems, and so on). One of the first attempts was based on the well-known expectation-maximization (EM) algorithm [22]. Since then, other frameworks based on gradient methods [23], [24] or the sum-product (SP) algorithm [12], [25] have been proposed in the literature.

The aim of this paper is to summarize and review the most recent contributions in the field of CA iterative synchronization. To do this, we will review in the next section a general model of a data-modulated signal embedding some unknown synchronization parameters to be estimated (carrier frequency and phase, symbol timing, etc.). We will stick for simplicity to the elementary case of constant parameters on the estimation interval (i.e., the data burst), and we will introduce on top of that problem the issue of iterative CA synchronization. After this is accomplished, we will fully develop the issue of CA synchronization in Section III, that represents in a sense the "core" of this paper. In particular, we will show how to develop CA turbo-synchronization algorithms based on a rigorous theoretical framework. Some examples of application of such techniques to data communication will be given in Section IV where we will compare the performance of turbo synchronizers with that of conventional DA and NDA schemes, and with the ultimate accuracy that can be attained by any synchronizer, namely, the Cramér–Rao bound (CRB) [2]. In the last section, we will draw some conclusions and we will mention some avenues for future research in this field, such as the extension of turbo synchronization to the tracking of time-varying parameters.

II. THE BASICS OF TURBO SYNCHRONIZATION

A. The Synchronization Problem for Gaussian Noisy Channel

The goal of any communication system consists in properly conveying some messages from a point (the transmitter side) to another (the receiver side). Typically, the messages to be transmitted have the form of a sequence of bits, say $\mathbf{u}^T = (u_0, u_1, \cdots, u_{L-1})$. In order to protect these messages of "information" against the noisy nature

of the *channel*,[1] *coding* is commonly used. It consists of adding some structured redundancy among the transmitted data. In this paper, we will assume that the coding operation is performed in two steps. First the information sequence is coded using a binary coder with rate[2] r, leading to a sequence $\mathbf{c}^T = (c_0, c_1, \cdots, c_{N-1})$ of $N = L/r$ coded bits, with $r \leq 1$. Then, in a second step, we assume that the coded sequence $\mathbf{c}$ is transformed into a sequence, say $\mathbf{a}^T = (a_0, a_1, \cdots, a_{K-1})$, of symbols belonging to an alphabet $\mathcal{A}$ of size M. For a particular choice of the binary code and the constellation alphabet $\mathcal{A}$, we will denote $\mathcal{S}$ the set of all possible sequences $\mathbf{a}$.

In order to be sent through the physical medium (free space, optic fiber, wired lines, ...), the sequence $\mathbf{a}$ has to be *modulated*, i.e., converted into an analog form. The reverse operation is the demodulation. It consists in the transformation of the received analog signal to discrete-time samples which can be further processed by some numerical algorithms. In practice, the modulation operations (and the associated demodulation operations) depends on the kind of channel we are facing. In this paper, for the sake of keeping things as simple as possible, we will focus on the simple model of bandpass transmission over an additive-white-Gaussian-noise (AWGN) channel. In this particular case, the baseband model of the received signal may be rewritten as

$$r(t) = A \sum_{k=0}^{K-1} a_k g(t - kT - \tau) e^{j(2\pi\nu t + \vartheta)} + w(t) \qquad (1)$$

where T is the symbol period, $g(t)$ is an analog pulse, $w(t)$ is an additive Gaussian noise with (known) power spectral density $\mathcal{N}_0$, and A, τ, ν, ϑ are unknown synchronization/channel parameters : amplitude, delay, carrier frequency offset, and carrier phase offset, respectively. In order to ease notation, we will collect the unknown parameters into a vector $\mathbf{b}^T = [A, \tau, \nu, \theta]$. Although our signal model is admittedly simplistic, the concepts and techniques that will be introduced in the sequel can be applied to different (more involved) contexts, such as multicarrier or multiple-input multiple-output (MIMO) systems, frequency-selective wireless channels, etc.

At the receiver, one has to apply some processing to the observed signal $r(t)$ in order to recover the transmitted message $\mathbf{u}$. First, the signal (1) is digitized at the sampling rate $1/T_s$, and the relevant signal samples are collected into the observation vector $\mathbf{r}$. From this vector one then computes an estimate, say $\hat{\mathbf{u}}$, of the information bits $\mathbf{u}$. The receiver making the best estimate [in the sense of minimizing the bit-error rate (BER)] decides on the transmitted bits u_l, $0 \leq l \leq L-1$, according to the maximum *a posteriori* (MAP) rule

$$\hat{u}_l = \arg\max_{\tilde{u}_l} \{p(\tilde{u}_l|\mathbf{r})\} \qquad (2)$$

where the *a posteriori* probability (APP) $p(u_l|\mathbf{r})$ is the marginal of the joint APP $p(\mathbf{u}, \mathbf{b}|\mathbf{r})$, i.e.,

$$p(u_l|\mathbf{r}) = \int \left[\sum_{\mathbf{u}\setminus\{u_l\}} p(\mathbf{u}|\mathbf{r}, \mathbf{b}) \right] p(\mathbf{b}|\mathbf{r}) d\mathbf{b}. \qquad (3)$$

We used the notation $\sum_{\mathbf{u}\setminus\{u_l\}}$ to indicate a summation over all the variables in $\mathbf{u}$ but u_l. Considering the presence of the channel code, evaluation of (3) is rather unfeasible, due both to the integration over the synchronization parameters $\mathbf{b}$, and also to the summation over the 2^{L-1} combinations of the transmitted bits $u_0, u_1, \cdots, u_{l-1}, u_{l+1}, \cdots, u_{L-1}$. Accordingly, our problem can reasonably be solved in a few specific cases only. In particular, if a "genie" provides the receiver with perfect prior knowledge of $\mathbf{b}$, i.e., $p(\mathbf{b}) = \delta(\mathbf{b} - \mathbf{b}_0)$, where $\delta(\cdot)$ is the Dirac function and $\mathbf{b}_0$ the actual parameter value, we have from (3)

$$p(u_l|\mathbf{r}) = \sum_{\mathbf{u}\setminus\{u_l\}} p(\mathbf{u}|\mathbf{r}, \mathbf{b}_0) = p(u_l|\mathbf{r}, \mathbf{b}_0) \qquad (4)$$

that means carrying out the function of *channel decoding* only. This can be efficiently accomplished (possibly with a certain loss of optimality) by well-known (possibly iterative) algorithms, based on the sum-product (SP) [26], [27] algorithm.

In practice, perfect *a priori* knowledge of the synchronization parameters is not available. As a consequence, we have to resort to some approximations of the optimal receiver. For example, in some contributions, see, e.g., [12] and [28], the authors propose to modify the metrics of the decoder in order to take the synchronization-parameter uncertainty into account. In this paper, we will stick to a simpler approach. We assume that $p(\mathbf{b}|\mathbf{r})$ is well approximated as

$$p(\mathbf{b}|\mathbf{r}) \simeq \delta\Big(\mathbf{b} - \hat{\mathbf{b}}(\mathbf{r})\Big) \qquad (5)$$

where $\hat{\mathbf{b}}(\mathbf{r})$ is a value that is derived from the observation vector (in the sequel, we will use the shorthand notation $\hat{\mathbf{b}} \triangleq \hat{\mathbf{b}}(\mathbf{r})$ for the sake of conciseness). This is tantamount to saying that the pdf function $p(\mathbf{b}|\mathbf{r})$ is concentrated around a given point $\hat{\mathbf{b}}$ or, in other words, that the amount of information contained in the received observations enables us to remove most of the uncertainty about $\mathbf{b}$. At

[1] I.e., the physical medium between the transmitter and the receiver.
[2] We see that r is a measure of the amount of redundancy added by the coder.

first sight, approximation (5) may appear quite strong; however, it is often valid in practice, and gives

$$p(u_l|\mathbf{r}) \approx \sum_{\mathbf{u}\setminus\{u_l\}} p(\mathbf{u}|\mathbf{r},\hat{\mathbf{b}}) = p(u_l|\mathbf{r},\hat{\mathbf{b}}). \tag{6}$$

Comparing (4) and (6), we see that approximation (5) leads to a well-known strategy in digital receivers which consists in first obtaining an estimate $\hat{\mathbf{b}}$ of the synchronization parameters, followed by decoding as if $\mathbf{b}$ was known and equal to $\hat{\mathbf{b}}$. Task of computing $\hat{\mathbf{b}}$ is just what we call *synchronization*.

B. The Synchronization Criterion

The first step in the design of a synchronizer is the choice of the criterion to derive our estimate $\hat{\mathbf{b}}$. Since the ultimate goal of the receiver is to minimize the error rate, the choice of $\hat{\mathbf{b}}$ should, in principle, be made in that sense. In practice, however, this solution is too complex. An alternative approach is thus computing $\hat{\mathbf{b}}$ according to a specific criterion taken from estimation theory. A popular criterion is the mean square estimation error (MSEE) minimization [29], [30]. Estimates (asymptotically) satisfying this criterion may be computed via the MAP estimator, i.e.,

$$\hat{\mathbf{b}} = \arg\max_{\tilde{\mathbf{b}}} \{\log p(\tilde{\mathbf{b}}|\mathbf{r})\}. \tag{7}$$

In the absence of any *a priori* side-information, (7) reduces to the simpler maximum-likelihood (ML) estimation rule

$$\hat{\mathbf{b}} = \arg\max_{\tilde{\mathbf{b}}} \{\log p(\mathbf{r}|\tilde{\mathbf{b}})\}. \tag{8}$$

Due to its good asymptotic properties, the ML (MAP) criterion is often considered in practice. Another key point of the success of the ML criterion is that, unlike other criteria, it straightforwardly leads to estimators that are not restricted to be linear. This is obviously a desirable feature in the context of synchronization where the observations depend in a nonlinear way on the synchronization parameters.

C. Synchronization Modes

Another key point in design of an estimator is the choice of the statistical information which will indeed be taken into account in the computation of $\hat{\mathbf{b}}$. In the context of ML estimation, the function to be maximized may be rewritten as

$$p(\mathbf{r}|\mathbf{b}) = \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b})p(\mathbf{a}). \tag{9}$$

It is clear from (9) that the computation of $p(\mathbf{r}|\mathbf{b})$, and consequently the implementation of the ML estimator, requires the knowledge of *i)* the noise distribution, i.e., $p(\mathbf{r}|\mathbf{a},\mathbf{b})$ and *ii)* the symbol sequence *a priori* knowledge, i.e., $p(\mathbf{a})$. When the transmission is coded, this *a priori* information has the following mathematical structure

$$\begin{aligned} p(\mathbf{a}) &= 1/|\mathcal{S}| \qquad \text{if } \mathbf{a} \in \mathcal{S} \\ &= 0 \qquad\quad\ \ \text{otherwise} \end{aligned} \tag{10}$$

where $\mathcal{S} \subset \mathcal{A}^K$ is set of possible coded sequences. The synchronizers which take (10) into account are said to operate in a CA mode. Unfortunately, the complexity required in taking the whole available statistical information into account may often turn out to be too large. For example, the evaluation of (9) for a nontrivial code is unfeasible due to the large number of terms in the sum. Hence, a solution to simplify the synchronizer implementation consists in dropping/approximating some parts of the available statistical information. Two particular approaches are commonly considered within the synchronization framework: the DA and the *non-code-aided* (NCA) modes.

The DA mode is intrinsically based on the fact that some symbols known to the receiver (pilots) have been inserted into the frame. We might call more properly this operating mode as *pilot-aided*. The DA approach consists in computing the estimate $\hat{\mathbf{b}}$ as if only the pilot symbols had been transmitted, i.e., by neglecting the received signal energy associated with the data symbols. Considering without loss of generality that the first P symbols in the data burst correspond to the pilots, the DA (pilot-aided) mode is characterized by the assumption that

$$p(\mathbf{a}) = \prod_{k=0}^{P-1} \delta(a_k - \breve{a}_k) \tag{11}$$

where $\breve{a}_k$ is the value of the pilot symbol at time k and where we let $K = P$. Since $p(\mathbf{a})$ is a Dirac function, the summation in (9) actually contains only one nonzero term and so closed-form expression of the (DA) ML estimator can be derived. In particular, we have [1], [2]

$$\begin{cases} (\hat{\nu}_{\mathrm{DA}}, \hat{\tau}_{\mathrm{DA}}) = \arg\max_{\tilde{\nu},\tilde{\tau}} \{|\Gamma(\tilde{\nu},\tilde{\tau})|\} \\ \hat{\vartheta}_{\mathrm{DA}} = \arg\{\Gamma(\hat{\nu}_{\mathrm{DA}}, \hat{\tau}_{\mathrm{DA}})\} \\ \hat{A}_{\mathrm{DA}} = \frac{|\Gamma(\hat{\nu}_{\mathrm{DA}},\hat{\tau}_{\mathrm{DA}})|}{\sum_{k=0}^{P-1} |\breve{a}_k|^2} \end{cases} \tag{12}$$

where

$$\Gamma(\tilde{\nu},\tilde{\tau}) \triangleq \sum_{k=0}^{P-1} \breve{a}_k^* x_k(\tilde{\nu},\tilde{\tau}) \tag{13}$$

and where $x_k(\tilde{\nu},\tilde{\tau})$ is the output at time kT of the filter matched to $g(t)$, assuming timing offset $\tilde{\tau}$ and frequency offset $\tilde{\nu}$, i.e.,

$$x_k(\tilde{\nu},\tilde{\tau}) \triangleq \int_{-\infty}^{\infty} r(t)e^{-j2\pi\tilde{\nu}t}g(t-kT-\tilde{\tau})dt. \qquad (14)$$

It follows from (13) that the estimates are computed from the function $\Gamma(\tilde{\nu},\tilde{\tau})$, which is obtained as the correlation of the pilot symbols $\breve{a}_k$ with the matched filter output samples $x_k(\tilde{\nu},\tilde{\tau})$.

When pilot symbols are not available, we change to NPA modes. In the past, the most common NPA mode was called *non-data-aided* (NDA) in the literature [1], [2]. Since it does not use any information about the possible channel code, we will call it *non-code-aided* (NPA-NCA or NCA for short). It is based on the assumption that

$$p(\mathbf{a}) = \frac{1}{|\mathcal{A}|^K} \quad \forall \mathbf{a} \in \mathcal{A}^K. \qquad (15)$$

In words, it assumes that all possible transmitted sequences are *a priori* equiprobable. This assumption is equivalent to dropping the statistical *a priori* information about the transmitted sequence: one transforms an informative prior (10) into a noninformative one, i.e., (15).

The conventional NDA synchronizers [1], [2] make use of the NCA mode. In addition, in order to cope with the huge summation in (9), these NDA synchronizers use an approximation for $p(\mathbf{r}|\mathbf{b})$ that gives rise to a closed-form expression of the summation (9). Two well-known synchronizers based on such an approach are the Viterbi&Viterbi carrier phase synchronizer [31] and the Oerder&Meyr timing synchronizer [32], which give an analytical solution for the (approximate) ML estimate of $\mathbf{b}$.

D. Lower Bounds on the Synchronization Performance

Until a few years ago, synchronizers were almost exclusively based on the (PA-)DA or the (NPA-)NCA modes [1], [2]. One of the motivations is that they bear a low implementation complexity (see Section II-E). However, as mentioned above, this reduction of the complexity involves a loss of statistical information: the DA synchronizer only considers the signal energy relative to pilots, see (12); the NCA mode does not take into account the *a priori* information available about the transmitted sequence, see (15).

In this context, we may ask the question of the degradation of the synchronizer performance due to these approximations. In order to answer this question, lower bounds on the MSEE have been proposed in the literature. In particular, the CRB [9], [10], [33], [34] and the modified CRB (MCRB) [35], [36] are good indicators of the best performance achievable by any estimator.

For simplicity, we consider the received signal $r(t)$ from (1), assuming that A, τ, and ν are known to the receiver; hence, only the carrier phase ϑ needs to be estimated. Formally, we have that the MSEE related to the estimation of a random uniformly distributed phase offset ϑ is lower bounded by the CRB, i.e., $E_{\mathbf{r},\vartheta}[(\vartheta-\hat{\vartheta})^2] \geq \mathrm{CRB}_\vartheta$, with

$$(\mathrm{CRB}_\vartheta)^{-1} = E_{\mathbf{r},\vartheta}\left[\left(\frac{d\log p(\mathbf{r}|\vartheta)}{d\vartheta}\right)^2\right] \qquad (16)$$

where $E_{\mathbf{v}}[.]$ denotes averaging with respect to $\mathbf{v}$. The extension related to the estimation of a vector parameter $\mathbf{b}$ can be found in [1], [2].

We see from (16) that the CRB is a function of the likelihood function $p(\mathbf{r}|\vartheta)$, which in turn is a function of $p(\mathbf{a})$ [see (9)]. We can therefore compute the CRB with different hypotheses about $p(\mathbf{a})$: (PA-)DA (11), (NPA-)NCA (15) and also NPA-CA with no pilots but with the aid of the code structure (10). The DA assumption enables the derivation of a closed-form expression of the CRB, i.e.,

$$\mathrm{CRB}_{\vartheta,DA} = \frac{1}{2P}\left(\frac{E_s}{\mathcal{N}_0}\right)^{-1} \qquad (17)$$

where E_s is the energy per transmitted symbol at the receiver input and P is the number of pilots. In both the NCA and CA scenarios, the evaluation of the CRB is considerably more difficult than in the DA scenario, because of the summation over $\mathbf{a}$ in (9). CRBs related to phase and timing estimation have been presented in [33] and [34] for the NCA scenario and in [9] and [10] for the CA scenario.

In order to avoid the complexity of (9) that is associated with the summation over $\mathbf{a}$, a *modified* CRB has been derived in [35], [36]: $E_{\mathbf{r},\vartheta}[(\vartheta-\hat{\vartheta})^2] \geq \mathrm{MCRB}_\vartheta$, with

$$(\mathrm{MCRB}_\vartheta)^{-1} = E_{\mathbf{r},\vartheta,\mathbf{a}}\left[\left(\frac{d\log p(\mathbf{r}|\mathbf{a},\vartheta)}{d\vartheta}\right)^2\right]. \qquad (18)$$

and where $\mathbf{a}$ is as in (15). When $p(\mathbf{r}|\mathbf{a},\vartheta)$ is Gaussian, the computational complexity related to evaluating the MCRB

is much smaller than with the CRB. The MCRB related to phase-offset estimation is easily obtained:

$$\mathrm{MCRB}_{\vartheta} = \frac{1}{2K}\left(\frac{E_s}{\mathcal{N}_0}\right)^{-1}. \tag{19}$$

Comparing (19) with (17), it follows that the MCRB equals the CRB that would be obtained if all symbols were known to the receiver.

The MCRB is in general looser than the CRB (i.e., $\mathrm{CRB}_{\vartheta} \geq \mathrm{MCRB}_{\vartheta}$), but is simpler to evaluate. It has been shown in [37] that for large $E_s/\mathcal{N}_0$ the CRB converges to the MCRB.

Fig. 1 shows the CRBs related to carrier phase estimation for the DA, NCA, and CA scenarios. We consider a burst of 100 QPSK symbols with convolutional encoding (constraint length 3, rates 1/2 and 1/5). In the DA scenario, five pilot symbols are transmitted. This amounts to a 5% content of pilots in the data burst, that represents a typical figure in digital communications. No pilot symbols are of course added in the (NPA) CA and NCA scenarios. In the NCA scenario, the synchronizer does not exploit the encoding rule, but assumes that all sequences of 100 QPSK symbols are equally likely (which is equivalent to uncoded QPSK transmission). Only in the CA scenario the encoding rule is taken into account when estimating the carrier phase. Also shown is the MCRB corresponding to 100 symbols [which is equivalent to a DA scenario where all 100 symbols are known to the receiver, see (17) and (19)].

It is clear from Fig. 1 that the performance of the DA mode is outperformed by the CA and NCA modes on a wide range of SNRs. This is due to the fact that the DA mode only allows to take benefit from the received signal energy associated with the pilots. From (17), we may notice that improving the performance of the DA synchronizer implies to increase either the number of pilots or the transmitted power. In practice, these solutions lead to losses in terms of power and spectral efficiency. Hence, the use of pilot should be avoided as much as possible.

Comparing the CA and NCA modes, we see that the corresponding CRBs are equal at sufficiently high SNRs (as predicted by [37]). In words, it means that operating in NCA or CA will lead to the same synchronization performance as long as the SNR is high enough. At lower SNRs, however, there is a gap between the performance that can be achieved by a NCA and a CA synchronizer. In our illustrative example, we see that the stronger the code, the larger the penalty when operating in NCA mode. As a consequence, CA synchronization has more and more become of crucial importance these recent years. Indeed, powerful error-correcting codes (such as turbo codes [6] and LDPC codes [7], [8]) allow state-of-the-art communication systems to operate very close to the channel capacity, i.e., at very small $E_s/\mathcal{N}_0$. It follows from Fig. 1 that NCA synchronizers might fail to provide reliable estimates when operating at such small $E_s/\mathcal{N}_0$. In such cases, CA estimators are needed to achieve accurate synchronization.

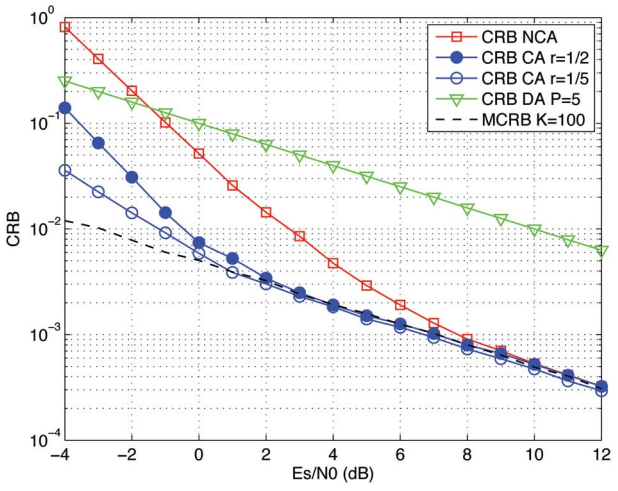


Fig. 1. ***CRBs associated with DA, NDA, and CA modes for carrier-phase synchronization and QPSK transmission.***

E. The Implementation: From Conventional to "Turbo" Synchronization

Once the modem designer has set the synchronization criterion (e.g., MMSE, MAP, ML...) and the synchronization mode (e.g., NCA, DA, CA...), s/he has to find a practical way to compute the estimate of the synchronization parameters.

Sometimes, especially in the DA mode, a simple closed-form expression of the ML estimator can be found. In other situations (typically in NPA-CA mode), it is unfortunately not feasible to do so. In such cases, an alternative solution consists in computing the estimate of the synchronization parameters by means of iterative methods. The conventional NPA *Decision-Directed* (DD) approach [1], [2] is an example of such an implementation. The principle of DD synchronizers is as follows. Starting from an initial estimate of the synchronization parameters, a decision is made about the transmitted symbols. This decision is in turn used to compute a new estimate of the synchronization parameters and so forth.[3] As an example, in the ML context a batch DD synchronizer would

[3]Note that the DD principle can be applied either in *batch mode*, i.e., by making a decision about all the symbols before recomputing a synchronization estimate, or in a *sequential mode*, i.e., the estimate of the synchronization parameters is updated after each new symbol decision.

compute a new estimate of the synchronizer parameter as follows:

$$\hat{\mathbf{b}}^{(n+1)} = \arg\max_{\tilde{\mathbf{b}}} p\left(\mathbf{r}|\hat{\mathbf{a}}^{(n)}, \tilde{\mathbf{b}}\right) \tag{20}$$

where $\hat{\mathbf{a}}^{(n)}$ is the symbol-sequence decision based on the previous estimate $\hat{\mathbf{b}}^{(n)}$. The DD approach can be used to implement CA and NCA synchronizers: if the symbol decisions are made by taking (10) into account, i.e., at the decoder output for example, the synchronizer will be CA; if the decisions are made assuming (15), the synchronizer is NCA.

Although quite straightforward to implement, the DD suffers from some drawbacks. First, the estimates computed from (20) do not necessarily converge to the actual ML solution (8). Second, the convergence of DD synchronizers is usually critical at low SNR: detected data are bad due to the low SNR, bad values are fed back into the synchronizer that fails to provide good values for the synchronization parameters, thus making the successively detected data even worse, and so on.

In order to attenuate this "hang-up" phenomenon occurring in the DD implementation, the concept of *Soft-Decision-Directed* (SDD) implementation has appeared in the literature, see, e.g., [18]. The principle is as follows: instead of computing the estimate of the synchronization parameters based on hard decisions, one considers *soft symbol decisions* carrying some measure of the confidence in the symbol decision. The intuitive idea behind the SDD approach is that less-reliable decisions should also have less "weight" in the synchronizer. The concept of SDD synchronization is a key ingredient of "*turbo synchronization*" since, in agreement with the Turbo Principle, it is based on the exchange of soft (rather than hard) information.

As such, the SDD implementation may appear as *ad hoc* as the DD implementation. In the sequel, we will however show that the (iterative) SDD approach can be cast in the framework of very well-known signal processing tools.

F. Turbo Synchronization: A First Naive Approach

Before formalizing a theoretical framework that will support the rigorous design of specific turbo-synchronization schemes (as detailed in Section III), we present now a simple, rather naive, example about the use of the soft information provided by an iterative channel decoder to derive a simple iterative SDD (ISDD) synchronization scheme. To keep our discussion as simple as possible, we assume that the only unknown synchronization parameter to be recovered is the carrier phase offset, and we focus on the approach suggested in [18] for a turbo-coded 16-QAM signal.

As mentioned in Section II-E, the idea of a (CA) SDD synchronizer would be to replace hard-detected data by soft symbol decisions that are derived from the decoder. As the computation of the soft decisions requires the availability of a phase estimate, the SDD synchronizer (just like the DD synchronizer) is iterative. According to this approach, the phase estimate at iteration l will have the form

$$\hat{\vartheta}^{(l)} = \arg\left\{\sum_{k=0}^{K-1} x_k(\nu,\tau)\lambda_k^{(l)*}\right\} \tag{21}$$

where $\lambda_k^{(l)*}$ is the soft symbol decision corresponding to symbol a_k.

In [18], the authors propose to compute the soft symbol decision $\lambda_k^{(l)*}$ as follows. First, they note that for a 16-QAM constellation, the symbol a_k can be represented as

$$a_k = c_k^{(1)}\left[2 + c_k^{(2)}\right] + jc_k^{(3)}\left[2 + c_k^{(4)}\right] \tag{22}$$

where $\{c_k^{(i)}, i = 1,2,3,4\} \in \{-1,1\}^4$ is the quadruplet of coded bits that are mapped onto a_k. As a consequence, finding a soft decision about a_k is immediate if we have soft decisions about coded bits $c_k^{(i)}$. Soft decisions about the coded bits can be derived by taking into account the principle of a turbo decoder: at each turbo iteration, the soft-in soft-out (SISO) modules in the turbo decoder exchange reliability metrics about the transmitted bits. In particular, the so-called logarithmic *a posteriori* probability ratios (LAPPR) $L_k^{(i,l)}$ are available at each iteration in the receiver. Denoting by $\hat{c}_k^{(i,l)}$ the soft decision about the bit $c_k^{(i)}$ at the l-th iteration, we can for example set $\hat{c}_k^{(i,l)} = \tanh(L_k^{(i,l)}/2)$ where

$$L_k^{(i,l)} = \log\left(\frac{p^{(l)}\left(c_k^{(i)} = 1|\mathbf{r}, \hat{\vartheta}^{(l-1)}\right)}{p^{(l)}\left(c_k^{(i)} = -1|\mathbf{r}, \hat{\vartheta}^{(l-1)}\right)}\right) \tag{23}$$

and $p^{(l)}(c_k^{(i)}|\mathbf{r}, \hat{\vartheta}^{(l-1)})$ is the approximation, at iteration l, of the actual *a posteriori* probability $p(c_k^{(i)}|\mathbf{r}, \hat{\vartheta}^{(l-1)})$ [26], [27]. The 16-QAM soft symbol $\lambda_k^{(l)}$ at iteration l is now formally coincident with a_k in (22), provided that $c_k^{(i)}$ is replaced with its soft counterpart $\hat{c}_k^{(i,l)}$. Note from (23) that the LAPPR to be used in the l-th iteration depends on the phase estimate obtained in the previous iteration. A few comments on this approach are in order.

- The tanh nonlinearity is the "soft" counterpart of a conventional threshold detector, and can take any value between -1 and $+1$.
- Unlike the *hard*-detected symbol provided by a conventional symbol-by-symbol detector or by a remapper, the *soft*-detected symbol does not necessarily coincide with a symbol of the constellation. In fact, the soft-symbol is close to a constellation point when the reliability is high,

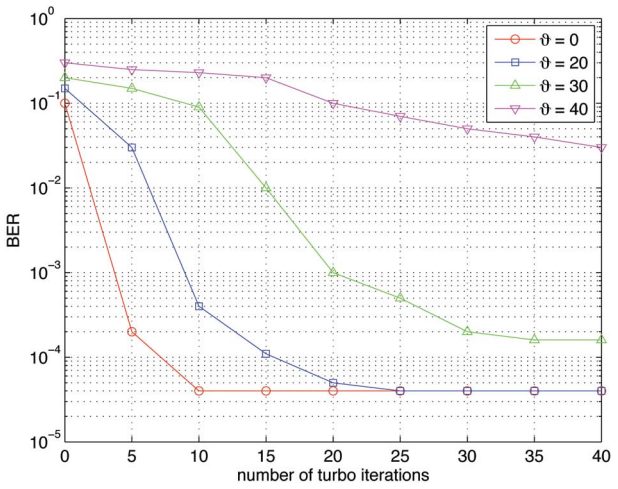


Fig. 2. ***BER of turbo-coded 16-QAM versus the number of turbo-decoder iteration at $E_b/\mathcal{N}_0$ = 6 dB for some values of the phase offset—rate-3/4 turbo parallel concatenated convolutional code with generators $g_1 = (\mathbf{31})_8$ and $g_2 = (\mathbf{33})_8$ and pseudorandom interleaver with length $L = 1500$.***

and to 0 if the reliability is low. This clearly seems beneficial to estimation because low-reliability terms in (21) are automatically weighted less.

- The *ad hoc* phase recovery algorithm illustrated so far is not optimal in the ML sense, although it can be loosely related to an ML criterion [18]. Nonetheless, we will show in Section III that one can end up to the same basic algorithm by means of a more rigorous approach.

To conclude, the (simulated) BER performance of the *ad hoc* turbo decoder/synchronizer scheme discussed above is illustrated in Fig. 2 at $E_b/\mathcal{N}_0 = 6$ dB as a function of the number of decoder iterations for some values of the phase offset ϑ, assuming an initial phase estimate $\hat{\vartheta}^{(0)} = 0$. The ISDD algorithm estimates the true phase offset without any degradation of the turbo-decoder performance compared to ideal phase recovery up to a phase error $|\vartheta| = 20°$, provided that the number of iterations can be extended up to 20. Conversely, for $|\vartheta| > 20°$ the convergence appears to be more critical, and some conventional algorithm for a rough initialization of the turbo synchronizer is requested.

III. MATHEMATICAL FRAMEWORKS FOR TURBO SYNCHRONIZATION

In the early days of turbo synchronization, the algorithms proposed in the literature were based on *ad hoc* reasoning and ingenuity, see, e.g., Section II-F. Although the efficacy of such approaches was soon recognized, a theoretical framework to justify their performance was missing. In particular, the following questions remained unanswered: *i*) Which is the ultimate performance one can expect from the synchronizer? *ii*) What kind of soft information (*a posteriori*, *extrinsic*, ...) should we use in the turbo synchronizer? *iii*) How should we exploit the available soft information to compute a "good" estimate of the synchronization parameters?

Since then, several frameworks for turbo synchronization have appeared in the literature. The common idea of these frameworks is to consider turbo synchronization as an iterative solution to the ML synchronization problem (8). Based on this approach, turbo synchronizers can be regarded as particular instances of powerful signal processing tools. In the remainder of this section, we will present some of the different turbo-synchronization frameworks which have been proposed so far in the literature. In particular, we consider frameworks based on the EM algorithm [22], [38], the gradient method [23], [24], [39], [40] and the sum-product (SP) algorithm [12], [25], [41].

A. Turbo Synchronization Based on the EM Algorithm

The EM algorithm [42] is an iterative method for solving ML problems. This algorithm proceeds in two steps: the expectation step (E-step) and the maximization step (M-step). At iteration $(n+1)$ we have

$$\text{E-step}: \mathcal{Q}\left(\mathbf{b}, \hat{\mathbf{b}}^{(n)}\right) = \int p\left(\mathbf{z}|\mathbf{r}, \hat{\mathbf{b}}^{(n)}\right) \log p(\mathbf{z}|\mathbf{b}) d\mathbf{z} \quad (24)$$

$$\text{M-step}: \hat{\mathbf{b}}^{(n+1)} = \arg\max_{\tilde{\mathbf{b}}} \mathcal{Q}\left(\tilde{\mathbf{b}}, \hat{\mathbf{b}}^{(n)}\right) \quad (25)$$

where $\hat{\mathbf{b}}^{(n)}$ is the estimate computed at the (previous) n-th iteration, and $\mathbf{z}$ is related to $\mathbf{r}$ by $\mathbf{r} = f(\mathbf{z})$, with $f(\cdot)$ denoting a many-to-one mapping. The actual observation set $\mathbf{r}$ and the extended observation set $\mathbf{z}$ are usually referred to as the *incomplete* and the *complete* data set, respectively. The so-called complete-data set, may actually be chosen in many different ways. In practice, however, some choices are more relevant than others. In particular, we may want to choose the complete-data set so that the M-step is easy to implement. Accordingly, $\mathbf{z}$ may often be thought as an (unavailable) observation vector such that the corresponding ML problem is "easy" to solve.

Like most iterative algorithms, the final convergence point of the EM algorithm depends on its initialization. In [43] it has however been shown that the fixed points of the EM algorithm must be stationary points of $\log p(\mathbf{r}|\mathbf{b})$. Note that the reverse is not true: all the stationary points of $\log p(\mathbf{r}|\mathbf{b})$ are not necessarily fixed points of the EM algorithm. Therefore, depending on its initialization, the EM algorithm may converge either to a saddle point, or to a local or the global maximum. In practice, the convergence to the global maximum is ensured as long as the initial estimate is "close enough."

Let us now apply the EM algorithm to our ML synchronization problem (8). In order to keep our subsequent discussions as simple as possible, we will concentrate on the case where all the synchronization parameters but the carrier phase offset ϑ are known. All conclusions we will draw are however valid in the general AWGN case. From our previous discussion in Section II, we know that the main problem in finding a tractable solution to the ML problem is the huge summation appearing in the expression of $p(\mathbf{r}|\mathbf{b})$, see (9). The EM algorithm alleviates this problem by breaking down the global maximization problem (8) into a sequence of easier problems, i.e., the M-steps (25), according to the choice of the complete data set. In the context of CA ML synchronization, the authors of [22] and [38] define the complete data set as $\mathbf{z} = (\mathbf{r}^T, \mathbf{a}^T)^T$. Using this definition, we have that the function that has to be maximized (with respect to ϑ) at each EM iteration is

$$\mathcal{Q}\left(\vartheta, \hat{\vartheta}^{(n)}\right) = \mathrm{Re}\left\{\sum_{k=0}^{K-1} \eta_k^*\left(\hat{\vartheta}^{(n)}\right) x_k(\nu,\tau) e^{-j\vartheta}\right\} \qquad (26)$$

where

$$\eta_k\left(\hat{\vartheta}^{(n)}\right) \triangleq \sum_{a\in\mathcal{A}} a\; p\left(a_k = a|\mathbf{r}, \hat{\vartheta}^{(n)}\right). \qquad (27)$$

Given the particular structure (26) of $\mathcal{Q}(\vartheta, \hat{\vartheta}^{(n)})$, the solution of the M-step is

$$\hat{\vartheta}^{(n+1)} = \arg\left\{\sum_{k=0}^{K-1} \eta_k^*\left(\hat{\vartheta}^{(n)}\right) x_k(\nu,\tau)\right\}. \qquad (28)$$

The EM synchronization framework represents a solution to the issues that we mention at the beginning of this Section: *i*) "turbo" synchronizers based on the EM algorithm achieve the ML performance as long as the EM algorithm is properly initialized; *ii*) the soft information handled by the synchronizer has to be delivered under the form of symbol *a posteriori* probabilities $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ [see (27)]; and *iii*) the new estimate of the phase offset has to be recomputed according to (28).

Note that the EM update (28) is very similar to the carrier-phase (PA-)DA-synchronizer (12). In fact the former is equal to the latter wherein the pilot symbols are replaced by "soft" symbol values η_k. This exactly corresponds to the intuitive approach we used in Section II-F to derive our carrier-phase turbo synchronizer. We can justify the "naive" turbo-synchronization approach presented in Section II-F by means of the EM framework, at least up to a point. In our naive approach, we used *approximations* of the coded-bit *a posteriori* probabilities instead of the actual symbol *a posteriori* marginals $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ as required by the EM algorithm, see (27).

The computation of symbol *a posteriori* probabilities $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ is a critical point in the implementation of the EM synchronizer as it represents the most complex operation. In some specific cases, these probabilities may be (exactly) computed with reasonable complexity by means of well-known algorithms. For example, in the case of BPSK convolutionally coded transmission, posterior probabilities $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ can be computed using the BCJR algorithm [44]. In other cases, the computation of such probabilities is far too complicated and suboptimal approaches are then considered. Turbo demodulation and turbo decoding are examples of such algorithms.[4] An exact EM synchronizer may therefore not be easily implemented in such receivers. In [38], a practical approximation of the EM synchronizer was proposed: the authors update the synchronization-parameter estimate at each turbo iteration by replacing in (27) the true *a posteriori* probabilities $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$ with their most recent approximation. Note that, making this approximation, the convergence properties of the EM algorithm are no longer ensured. In particular, such an algorithm is no longer ensured to reach the ML solution. In fact, it has recently been shown that, although the approximated EM algorithm is not ensured to converge, its fixed points must be stationary point of the Bethe approximation of the system free energy [45]. Despite of its suboptimality, the efficiency of this approach has nevertheless been shown in several contributions, see, e.g., [46] and [47].

B. Turbo Synchronization Based on Gradient Methods

In the previous section, we emphasized that a proper application of the EM algorithm to our synchronization problem naturally leads to a framework for turbo synchronization. A direct extension of this approach is to examine whether other iterative maximum-search algorithms could lead to other, different, solutions of the ML synchronization problem. In particular, we present in this section a synchronization framework based on gradient methods, see [23], [24], [39], [40]. Again, for the sake of clarity, we limit our discussion to the case of an unknown carrier phase offset.

Gradient methods are based on the simple fact that the likelihood function $\log p(\mathbf{r}|\mathbf{b})$ is ensured to locally increase in the "direction" of its gradient. Based on this idea, we can update our estimates as follows:

$$\hat{\mathbf{b}}^{(n+1)} = \hat{\mathbf{b}}^{(n)} + \alpha^{(n)} \left(\nabla \log p(\mathbf{r}|\mathbf{b})\right)_{\mathbf{b}=\hat{\mathbf{b}}^{(n)}} \qquad (29)$$

[4]These algorithms are actually particular instances of the sum-product algorithm applied on factor graphs with cycles [26], [27].

where $\alpha^{(n)} > 0$. This equation has an easy interpretation: we update the previous estimate $\hat{\mathbf{b}}^{(n)}$ in the direction $(\nabla \log p(\mathbf{r}|\mathbf{b}))_{\mathbf{b}=\hat{\mathbf{b}}^{(n)}}$ with an amplitude given by $\alpha^{(n)}$. Now, since $\log p(\mathbf{r}|\mathbf{b})$ is ensured to locally increase in the direction of its gradient, we know that there exists $\alpha^{(n)} > 0$ such that $\hat{\mathbf{b}}^{(n+1)}$ increases the likelihood function. The choice of the step factor $\alpha^{(n)}$ is therefore crucial for the convergence of gradient algorithms. There exist a number of rules for choosing the stepsize $\alpha^{(n)}$: the minimization rule, the Armijo rule, the Goldstein rule, etc. We refer the interested reader to [48].

Regarding the direction of update, there actually exist many other choices than $\nabla \log p(\mathbf{r}|\mathbf{b})$ which still ensure a local increase of the likelihood function. Different choices of the update direction lead to methods such as the conjugate-gradient methods, the Newton–Raphson methods [48], etc. In fact, some direction choices are more "suitable" in some problems than others and lead therefore to higher speed of convergence. In the remainder of this section, we will however restrict our discussion to the case of the steepest-descent (SD) methods described in (29) for the sake of conciseness.

We see from (29) that a crucial point in gradient methods consists in efficiently evaluating the gradient of $\log p(\mathbf{r}|\mathbf{b})$. In [10], [24], [39], and [40], the authors have derived expressions to do so. In particular, coming back to our tutorial model (1) and assuming that only ϑ is unknown, we have

$$\frac{\partial}{\partial \vartheta} \log p(\mathbf{r}|\vartheta) = \frac{2A}{\mathcal{N}_0} \mathrm{Im}\left\{ \sum_{k=0}^{K-1} \eta_k^*(\vartheta) x_k(\nu, \tau) e^{-j\vartheta} \right\} \quad (30)$$

where $\eta_k(\vartheta)$ is defined in (27). We see from (30) that, like the EM synchronizer, the SD-based approach only requires the knowledge of the symbol *a posteriori* marginals $p(a_k|\mathbf{r}, \mathbf{b})$ to compute the updated estimates of the synchronization parameters. Hence, CA synchronization by means of the SD algorithm can be implemented by iteratively exchanging information between the symbol-detection module (which computes probabilities $p(a_k|\mathbf{r}, \mathbf{b}^{(n)})$) and the synchronizer module [which computes $\hat{\mathbf{b}}^{(n+1)}$ according to (29) and (30)].

Unlike the EM algorithm, the set of fixed points of gradient methods contains *all* the stationary points of $\log p(\mathbf{r}|\mathbf{b})$. It is easy to see from (29) that a zero gradient implies $\hat{\mathbf{b}}^{(n+1)} = \hat{\mathbf{b}}^{(n)}$. Therefore, denoting by $\Gamma_{\mathrm{Gradient}}$ (resp. Γ_{EM}) the set of fixed of points of the gradient (resp. EM) synchronizers, we have

$$\Gamma_{\mathrm{EM}} \subseteq \Gamma_{\mathrm{Gradient}}. \quad (31)$$

This means that the gradient synchronizer is more "likely" than the EM synchronizer to get stucked to a point which is not the ML solution. However, like the EM algorithm, the convergence of the gradient method to the ML solution is ensured as long as it is initialized "close enough." Note moreover that the speed of convergence of the gradient methods may much larger than the EM algorithm in a number of situations [49].

Finally, notice that the implementation of gradient-based synchronizers requires to compute the symbol *a posteriori* marginal probabilities $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n)})$. The same discussion as the one made in Section III-A applies here: symbol *a posteriori* probabilities may be exactly evaluated in some specific cases; in other cases, we have to resort to approximations. In the approximated case, the fixed points of the gradient synchronizers are equal to the stationary points of the Bethe approximation of the free energy.

C. Turbo Synchronization Based on the SP Algorithm

In this section, we place turbo synchronization into the context of factor-graph representations and the sum-product (SP) algorithm [26], [27].

A factor graph is a bipartite graph which represents the factorization of a function. A factor graph 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 SP algorithm is a "message-passing" algorithm to be run on factor graphs and that allows efficient computation of the marginals of the function that the graph represents. If the graph has no cycles, the SP algorithm can compute all the marginals in a finite number of steps. Well-known algorithms, like the BCJR decoder for example, may be viewed as particular instances of the SP algorithm applied on graphs without cycles [26]. On the other hand, if the graph has cycles, message updates along cycles lead to an iterative algorithm with no natural termination. In this case, it can no longer be proved that the results delivered by the SP algorithm are exact. The SP algorithm for channel decoding may or may not perform well, depending on the chosen code structure, block length, SNR, etc. The "turbo" decoder is an example of the SP algorithm applied on cyclic factor graphs.

We intend to show now how to derive turbo-synchronization methods based on the SP-algorithm framework. Before entering into deeper considerations, let us sketch the main ideas behind the SP-algorithm synchronization. First, remember that the synchronization operation is a direct consequence of an approximation of the bitwise optimal receiver (see Section II-A). Now, the implementation of the optimal receiver (which requires the computation of marginals $p(u_l|\mathbf{r})$) may actually be placed into the SP-algorithm framework. As we will see in the remainder of this section, applying the same kind of approximation as (5) to the SP-implementation of the optimal receiver will lead to new turbo-synchronization algorithms.

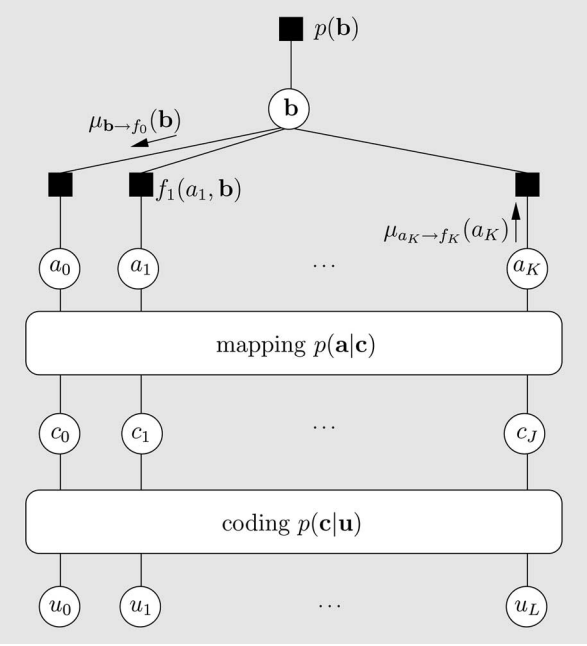


Fig. 3. ***Factor-graph representation of*** $p(\mathbf{r},\mathbf{a},\mathbf{c},\mathbf{u},\mathbf{b})$***. The box*** $p(\mathbf{a}|\mathbf{c})$ ***represents the factor graph relative to bit-to-symbol mapping. The box*** $p(\mathbf{c}|\mathbf{u})$ ***represents the factor graph characterizing the error-correcting code used for the transmission. Functions*** $f_k(a_k,\mathbf{b})$ ***are such that*** $p(\mathbf{r}|\mathbf{a},\mathbf{c},\mathbf{u},\mathbf{b})=\prod_{k=0}^{K-1} f_k(a_k,\mathbf{b})$**.**

First, notice that marginals $p(u_l|\mathbf{r})$ required to implement the bitwise optimal receiver (2) may be computed by applying the SP algorithm on a factor graph representation of $p(\mathbf{r},\mathbf{a},\mathbf{c},\mathbf{u},\mathbf{b})$ (see Fig. 3). Unfortunately, the implementation of the SP algorithm on the factor graph in Fig. 3 is often cumbersome because one has to handle messages $\mu_{\mathbf{b}\to f_k}(\mathbf{b})$ of continuously variable $\mathbf{b}$. In order to deal with this difficulty, an approach first proposed in [50] consists of approximating $\mu_{\mathbf{b}\to f_k}(\mathbf{b})$ by a message $\tilde{\mu}_{\mathbf{b}\to f_k}(\mathbf{b})$ which can be characterized by a finite number of parameters. The simplest example of such an approach is

$$\tilde{\mu}_{\mathbf{b}\to f_k}(\mathbf{b}) = \delta(\mathbf{b}-\hat{\mathbf{b}}) \quad \forall k \tag{32}$$

where all messages $\tilde{\mu}_{\mathbf{b}\to f_k}(\mathbf{b})$ are fully characterized by the value of $\hat{\mathbf{b}}$. This approach has been followed in [12], [25], [41]. Based on some considerations about the messages exchanged by the SP algorithm, the authors of these papers compute $\hat{\mathbf{b}}$ according to

$$\hat{\mathbf{b}} = \arg\max_{\mathbf{b}} \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a},\mathbf{b}) \prod_k \mu_{a_k\to f_k}(a_k). \tag{33}$$

Let us pause a moment to compare (33) with (8) and (9). We see that the function maximized in (33) has the same structure as a likelihood function wherein the data symbols are assumed to be distributed according to $\prod_k \mu_{a_k\to f_k}(a_k)$. From this observation, we see that (33) can be understood as a modified ML problem in which SP messages $\mu_{a_k\to f_k}(a_k)$ are used as *a priori* information on the transmitted symbols.

In the general case, the factor graph contains cycles and the SP algorithm is therefore iterative. A new estimate $\hat{\mathbf{b}}^{(n)}$ may then computed at each iteration according to (33). This kind of synchronizers has been first proposed based on intuitive reasoning in [19] and later justified throughout the SP framework in [12], [25], and [41].

A direct consequence of the observation that (33) has the structure of an ML problem is that synchronization frameworks presented in Sections III-A and B may be used to computed the value which satisfies the modified maximization problem (33). In fact, all the relevant equations still hold provided that we replace $p(\mathbf{a})$ with $\prod_k \mu_{a_k\to p}(a_k)$. In [25] and [41], for example, the authors propose to apply the EM algorithm to (33). In [12], [24], and [39], the authors apply gradient methods to solve the modified ML problem. Note that, since both the SP and the EM (resp. gradient) algorithms are iterative, the combination of these algorithms leads to a *doubly iterative* synchronization procedure. In a first step, SP messages $\mu_{a_k\to f_k}(a_k)$ are computed using the previous estimate $\hat{\mathbf{b}}^{(n-1)}$. Then, a new estimate $\hat{\mathbf{b}}^{(n)}$ is computed by solving (33) via an EM (resp. gradient) algorithm. Some comments about this approach are as follows.

- The complexity associated with one iteration of the EM/gradient algorithm is usually much smaller [25] when considering the maximization problem (33) than when considering the maximization of $p(\mathbf{b}|\mathbf{r})$. This is due to the fact that $\prod_k \mu_{a_k\to f_k}(a_k)$ has a nice factorization whereas actual prior $p(\mathbf{a})$ does not necessarily factor [12], [41].
- The number of EM/gradient iterations performed to solve (33) may be tuned according to system requirements. In particular, it is emphasized in [25], [39], and [41] that if only one single EM (resp. gradient) iteration is performed at each SP iteration, the resulting synchronizer is equivalent[5] to the EM (resp. gradient) synchronizer presented in Section III-A.

Before concluding this section, let us discuss the important question of the fixed points of the SP synchronizer (33). In [41], it is emphasized that the EM synchronizer may be regarded as a particular case of the SP

[5]The resulting synchronization algorithm is equivalent to an exact EM synchronizer if the factor graph on which is based the computation of $\mu_{a_k\to f_k}(a_k)$ is cycle-free and approximated otherwise (see Section III-A).

synchronizer (when the modified ML problem (33) is solved via an EM algorithm). As a direct consequence, it turns out that any fixed point of the SP synchronizer must also be a fixed point of the EM synchronizer, the reverse being not necessarily true. Denoting by Γ_{SP} the set of stationary points of the SP synchronizer, we have therefore

$$\Gamma_{\mathrm{SP}} \subseteq \Gamma_{\mathrm{EM}} \subseteq \Gamma_{\mathrm{Gradient}}. \tag{34}$$

The SP synchronizer seems therefore to further restrict the set of fixed points at which the synchronizer can be stucked. Let however make two important remarks on (34). First, the set ordering (34) only holds for "pure" SP synchronizer as described in (33), i.e, if we indeed compute the solution of (33) at each iteration. In particular if we use an EM (resp. gradient) algorithm to solve (33), it is straightforward from our second remark above that $\Gamma_{\mathrm{SP}} = \Gamma_{\mathrm{EM}}$ (resp. $\Gamma_{\mathrm{SP}} = \Gamma_{\mathrm{Gradient}}$). Secondly, note that unlike the EM and gradient synchronizers, the SP synchronizer is *not* ensured to converge to its fixed points.

When the factor graph on which are based the computation of messages $\mu_{a_k \to f_k}(a_k)$ contains cycles, the result mentioned above may be translated in terms of Bethe approximation of the system free energy. In particular, the fixed points of the SP synchronizer must be stationary point of the Bethe approximation of the free energy.

IV. APPLICATIONS AND PERFORMANCE ANALYSIS

In this section, we illustrate, by simulation results, the performance of the turbo synchronizers in terms of *i)* their *accuracy*, i.e., the MSEE they can achieve; *ii)* their *acquisition* or *operating range*, i.e., the range of parameter values for which they properly recover synchronization; and *iii)* their *impact on the BER performance* in synchronizing state-of-the-art receivers.

A. Accuracy Evaluation

We first compare the MSEE achieved via turbo synchronization (after convergence) with the MSEE achieved by conventional NCA synchronizers (see Section II-C). For the sake of illustration, we consider a Gray-mapped QPSK convolutionally coded transmission. In such a scenario, the symbol *a posteriori* probabilities may be computed exactly by means of a BCJR algorithm. As a consequence, the EM and the SD synchronizers, respectively described in Sections III-A and B, can be implemented exactly. It can also be shown [39] that, in such a situation, the factor graph associated with the computation of messages $\mu^{(n)}_{a_k \to p}(a_k)$ (see Section III-C) is cycle-free. Hence, the fixed points of the EM, SD and SP synchronizers must be stationary points of $p(\mathbf{r}|\mathbf{b})$.

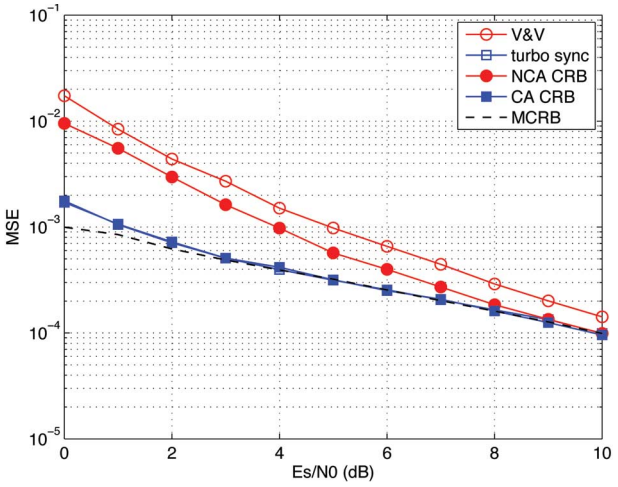


Fig. 4. ***Phase estimation: MSEE versus $E_s/\mathcal{N}_0$-ratio for a QPSK convolutionally coded transmission.***

Fig. 4 depicts the MSEE achieved by different carrier-phase synchronizers. The channel code is a rate-1/2 nonsystematic convolutional code with generator polynomials $[5, 7]_8$ [51]. The transmitted frames consist of 500 QPSK symbols. We have considered: *i)* the conventional NCA Viterbi&Viterbi carrier phase synchronizer [31] and ii) the EM, SD and SP turbo synchronizers. The turbo synchronizers have been initialized "close"[6] to the global maximum and run until convergence. Their performance is similar and has been illustrated by one single curve. For the sake of comparison with the synchronizer performance, the CRB (in both the NCA and the CA scenarios) and the MCRB have also been represented.

We see that the turbo synchronizers attain the CA CRB. This is a direct consequence of the fact that, as shown in Section III, turbo synchronization is an iterative implementation of the ML CA estimator. Since the ML estimator is (asymptotically) efficient [29], i.e., reaches the CRB for sufficiently long frame lengths, we may expect the turbo synchronizers do so if they are properly initialized. On the contrary, we may notice that the MSEE achieved by the Viterbi&Viterbi synchronizer does not exactly attain the NCA CRB. This is due to the fact that the Viterbi&Viterbi synchronizer is only an approximation of the ML criterion (see Section II-C). The reason of the gap between the performance achievable by turbo synchronizers and the conventional V&V synchronizer is therefore twofold: *i)* the Viterbi&Viterbi operates in a NCA mode and drops therefore important statistical information about the transmitted sequences and *ii)* the V&V does not implement the actual NCA ML solution but is based on an approximation.

[6]By "close," we mean at a value such that any problem of misconvergence is avoided.

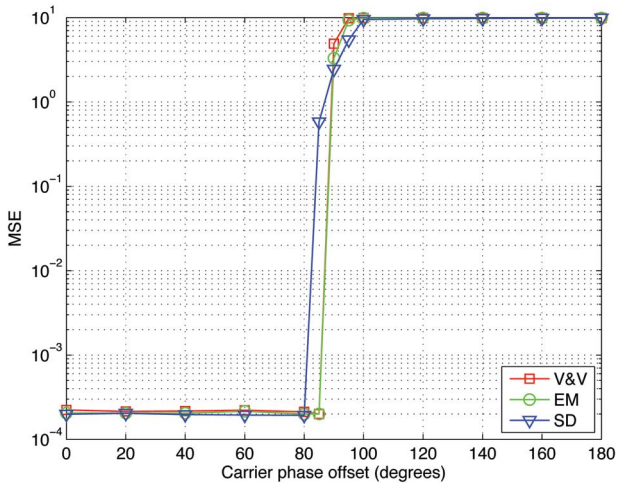


Fig. 5. *MSEE versus the phase offset for different synchronizers.*

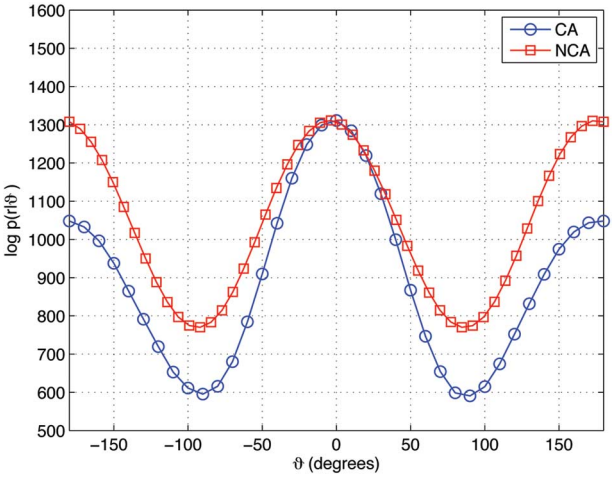


Fig. 6. *One realization of the log-likelihood function* $\log p(\mathbf{r}|\vartheta)$ *in the NCA and the CA cases.*

B. Operating Range

Another important feature of any synchronizer is its ability to attain synchronization for a large range of synchronization-parameter values, i.e., its *acquisition range*. For the sake of illustration, we consider the following scenario: phase synchronization of a BPSK convolutionally coded transmission with $\vartheta = 0$.

We have illustrated in Fig. 5 the MSEE achieved by different synchronizers versus the actual phase offset. All the turbo synchronizers are initialized at $\hat{\vartheta}^{(0)} = 0°$. We may notice that the turbo (EM and SD) synchronizers exhibit the same acquisition range as the V&V NCA synchronizers, namely both of them are unable to properly synchronize the system when the absolute value of the phase offset is larger than 90°. The justification of such a behavior is however quite different in the conventional (NCA) and the turbo (CA) scenarios: *i)* the V&V is a NCA synchronizer and has its (phase-offset) acquisition range intrinsically limited by the constellation symmetry [1] and *ii)* the turbo-synchronizer acquisition range is limited due to convergence failure.

Fig. 6 illustrates these considerations: we have represented the log-likelihood function (LLF) $\log p(\mathbf{r}|\mathbf{b})$ in both the NCA and the CA scenarios. We first notice that the NCA LLF has a 180° symmetry.[7] This symmetry implicitly means that, irrespective of its implementation, the NCA synchronizer will not be able to distinguish between two phase offsets that are 180° apart. As a direct consequence, NCA synchronizers, like the V&V, are unable to attain synchronization for phase offsets $|\vartheta| > 90°$ unless the ambiguity is somehow solved. On the other hand, we may notice that the CA log-likelihood function exhibits one single global maximum in the range $[-180°, 180°]$. In the considered scenario,[8] the CA ML synchronizer appears to be able to get rid of the 180° ambiguity typical of NCA synchronizers. However, the turbo synchronizers described in Section III are based on iterative maximum-search methods, and we see that the CA LLF in Fig. 6 has additional local maxima at $\pm 180°$. If the algorithm is initialized at $|\hat{\vartheta}^{(0)}| > 90°$, the "hill-climbing" nature of iterative methods (i.e., the EM algorithm, the gradient methods...) on which the turbo synchronizers rely will lead them to converge to a local maximum rather than to the global one. Hence, although having the necessary statistical information at their disposal, the iterative nature of the turbo synchronizer makes them unable to recover the correct synchronization on the whole range $[-180°, 180°]$ as well. In such cases, ambiguity resolution methods have to be considered, see, e.g., [53]–[55] and references therein. Another approach consists of using codes robust to a 180°-rotation [20], [52]. Using such codes, the FER-BER performance achieved by the system are identical irrespective of the maximum (local or global) to which converges the synchronizer.

C. Synchronization of State-of-the-Art Receivers

In this section, we illustrate the impact of turbo synchronization algorithms on the performance achieved by state-of-the-art receivers. Admittedly, not all data modems require turbo synchronization. In some cases, conventional synchronization methods will just perform as fine and will be sufficient for properly locking the received signal. In other cases, the use of more powerful synchronizers is required. In practice, the choice of a

[7]It is easy to show that this symmetry is actually due to the NCA nature of the synchronizer [1], [2].

[8]Depending on the considered code, the log-likelihood function may or may not exhibit a global maximum [39]. For example "noncoherent" codes (i.e., codes enabling to recover the transmitted sequence under a 180°-rotation [20], [52]), exhibits a CA LLF similar to the NCA LLF due to the problem symmetry.

proper synchronization method has to be made on a case-by-case basis.

Our case study will be here the joint phase and timing synchronization of a turbo-coded transmission. The considered turbo encoder consists of the parallel concatenation of two rate-1/2 recursive systematic convolutional (RSC) encoders with polynomial generators $(37, 33)_8$, separated by a random interleaver. The turbo-encoder output is punctured so that the overall code rate is 1/2. Transmitted frames consist of 999 BPSK symbols. The rolloff factor is set to 0.1. The timing and the phase offset are changed for each new transmitted frame and are assumed to be uniformly distributed respectively on $[-T/2, T/2]$ 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 $|\tau - \hat{\tau}| \leq T/2$ and $|\vartheta - \hat{\vartheta}| < \pi/2$.

Figs. 7 and 8 represent the BER achieved by the receiver versus the $E_b/\mathcal{N}_0$-ratio and the number of required operations, respectively. We have considered the V&V and Oerder&Meyr (O&M) [1] conventional NCA synchronizers and the EM, SD and SP turbo synchronizers described in Section III. In the SP case, we consider the computation of the maximum of $\log p^{(n)}(\mathbf{r}|\mathbf{b})$ according to EM and SD algorithms. The resulting synchronizers are respectively denoted SP-EM(n) and SP-SD(n), where n is the number of EM (resp. SD) iterations performed per SP (or equivalently turbo) iteration. As mentioned in Section III-C, the approximated versions of the EM and the SD synchronizers are respectively equivalent to SP-EM(1) and SP-SD(1), and will be referred to as such in the sequel. All the iterative synchronizers are initialized by means of the V&V and O&M NCA synchronizers.

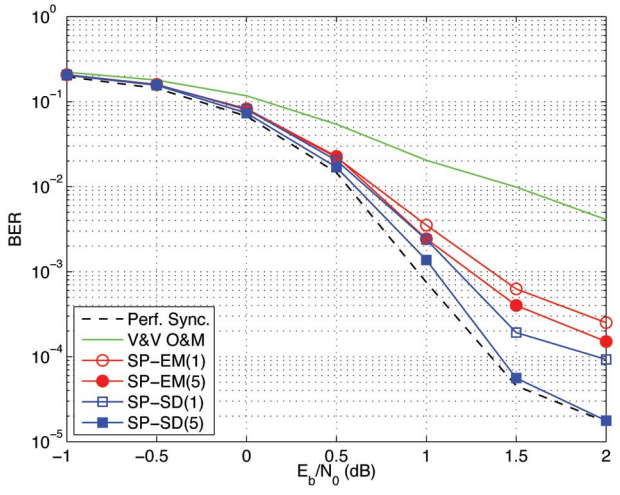


Fig. 7. ***BER achieved at the 15th turbo iteration with different synchronization algorithms.***

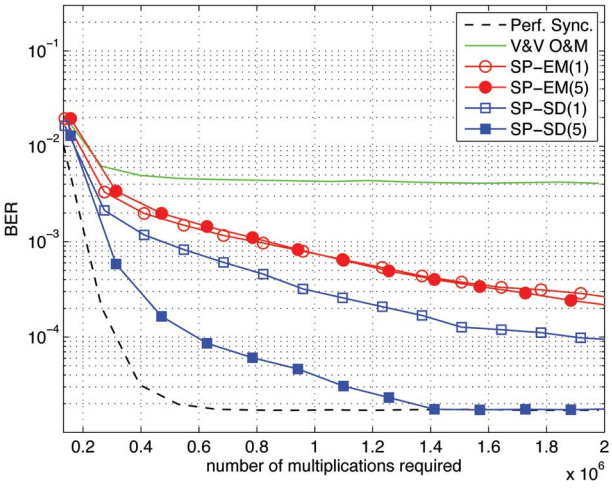


Fig. 8. ***BER versus the number of required operations at*** $E_b/\mathcal{N}_0 = 2$ ***dB.***

Fig. 7 illustrates the BER versus $E_b/\mathcal{N}_0$ after 15 turbo iterations. We note that the conventional synchronizers lead to dramatic BER degradations, while the turbo synchronizers enable to improve the BER achieved by the system. In particular, we see that, in the considered scenario, the SP-SD(5) synchronizer enables to recover the performance of the perfectly synchronized system. Hence, it is clear from this example that turbo synchronization is beneficial to improve system performance.

As far as the considered setup is concerned, it seems that increasing the number of EM (resp. SD) iterations at each SP iteration improves the final BER, i.e., the SP-EM(5) [resp. SP-SD(5)] leads to better performance than the SP-EM(1) [resp. SP-SD(1)]. The increase of the number of iterations however implies a higher computational cost per turbo iteration. In this context, we may ask the question of the turbo-synchronization method leading to the best tradeoff between BER and complexity. In order to have a measure of the computational complexity, we have represented in Fig. 8 the BER versus the number of operations[9] required by each method. Each point corresponds to an (additional) "decoding-synchronization" iteration. For the sake of comparison, the BER of both the perfectly synchronized system and the system synchronized via conventional NCA synchronizers has also been plotted. We first note that the system synchronized by means of the V&V and the O&M synchronizers can no longer improve the BER after a few iterations. This is due to the bad estimation quality of these synchronizers at such a low SNR. On the contrary, the systems based on turbo synchronization keep on improving the BER throughout the iterations. The computational complexity associated with this improvement is however quite different depending on the

[9]We consider the (approximated) number of multiplications required by the system.

considered turbo-synchronization method. As far as our simulation setup is concerned, the SP-SD(5) turbo synchronizer exhibits the best tradeoff between performance and complexity. Also, notice that increasing the number of SD iterations per SP iteration reduces the system complexity: achieving a BER of 10^{-4} requires about three times more multiplications when the system is synchronized via the SP-SD(1) than when using the SP-SD(5). This is not the case for the SP-EM synchronizer: both the SP-EM(1) and the SP-EM(5) exhibit the same "rate" of improvement. In a general way, it is not possible to draw conclusions about the turbo-synchronization method leading to the best "BER-complexity" tradeoff. As mentioned at the beginning of this section, the "best" synchronization method is a function of the considered setup and has to be made on a case-by-case basis.

V. CONCLUSIONS AND PERSPECTIVES

The paper has illustrated a few techniques to perform joint iterative detection and synchronization, i.e., *turbo-synchronization* in digital modems. Turbo synchronization has been emphasized to be a suitable implementation when synchronizer is wished to operate in CA mode (i.e., when it takes the code structure into account in the synchronization process).

In the authors' opinion, CA turbo synchronization has to be adopted whenever possible in modem design to make data detection more robust. This specifically applies when the SNR is low and/or when the number of pilot symbols the receiver can rely on is limited. The complexity of such techniques, although nonnegligible with respect to conventional synchronization, turns out not to be crucial in the context of the whole modem design when compared to that of the channel decoder for iterative decoding. Attention has to be paid to the initialization of the iterative algorithm, but this issue can be usually solved with the aid of a very short preamble, on which traditional data-aided synchronization can provide a good coarse (initial) value of the parameter under estimation. Different approaches can then be adopted as far as the iterative algorithm is concerned (EM, gradient, and so on) depending on the required performance, and on the affordable complexity.

The paper has not touched upon a few topics that comes naturally to one's mind. First and foremost, we mention here the estimation through the same iterative techniques of time-varying signal parameters, such as the signal amplitude out of a time-selective wireless channel, or the phase noise process of a receiver equipped with a low-cost, consumer-grade oscillator. In such cases the issue of constant parameter estimation is turned into that of estimation of the realization of a *random process*. A few techniques such as forward-backward estimation on a code block of the random process (fading channel or phase noise), as well as augmentation of the sum-product decoding/estimation algorithm to perform the same function are exhibiting good results. And they all can be seen as belonging to an enlarged family of "extended turbo-synchronization" techniques. ■

REFERENCES

[1] H. Meyr, M. Moeneclaey, and S. Fechtel, *Digital Communication Receivers: Synchronization, Channel Estimation and Signal Processing*, ser. Telecommunications and Signal Processing. New York: Wiley, 1998.

[2] U. Mengali and A. N. D'Andrea, *Synchronization Techniques for Digital Receivers*. New York: Plenum, 1997.

[3] S. ten Brink, J. Speidel, and J. C. Yan, "Iterative demapping and decoding for multilevel modulation," in *Proc. IEEE Global Communications Conf. (GLOBECOM)*, 1998, pp. 579–584.

[4] A. Glavieux, C. Laot, and J. Labat, "Turbo equalization over a frequency selective channel," in *Proc. Int. Symp. Turbo Codes and Related Topics (ISTC)*, 1997, pp. 96–102.

[5] X. Wang and H. V. Poor, "Iterative (turbo) soft interference cancellation and decoding for coded CDMA," *IEEE Trans. Commun.*, vol. 47, no. 7, pp. 1046–1061, Jul. 1999.

[6] C. Berrou, A. Glavieux, and P. Thitimajsima, "Near Shannon limit error-correcting coding and decoding," in *Proc. IEEE Int. Conf. Communications (ICC)*, 1993, pp. 1064–1070.

[7] R. G. Gallager, "Low density parity-check codes," *IEEE Trans. Inf. Theory*, vol. IT-8, no. 1, pp. 21–29, Jan. 1962.

[8] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," IEEE Trans. Inf. Theory, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[9] N. Noels, H. Wymeersch, H. Steendam, and M. Moeneclaey, "Carrier and clock recovery in (turbo) coded systems: Cramer-Rao bound and synchronizer performance," *EURASIP J. Appl. Signal Process.*, vol. 2005, no. 6, pp. 972–980, May 2005.

[10] N. Noels, H. Steendam, and M. Moeneclaey, "The Cramer-Rao bound for phase estimation from coded linearly modulated signals," *IEEE Commun. Lett.*, vol. 7, no. 5, pp. 207–209, May 2003.

[11] A. Anastasopoulos and K. M. Chugg, "Adaptive iterative detection for phase tracking in turbo-coded systems," *IEEE Trans. Commun.*, vol. 49, no. 12, pp. 2135–2144, Dec. 1991.

[12] J. Dauwels and H.-A. Loeliger, "Phase estimation by message passing," in *Proc. IEEE Int. Conf. Communications (ICC)*, 2004, pp. 523–527.

[13] G. Colavolpe, G. Ferrari, and R. Raheli, "Noncoherent iterative (turbo) decoding," *IEEE Trans. Commun.*, vol. 48, no. 9, pp. 1488–1498, Sep. 2000.

[14] B. Mielczarek and A. Svensson, "Phase offset estimation using enhanced turbo decoders," in *Proc. IEEE Int. Conf. Communications (ICC)*, New York, 2002, pp. 1536–1540.

[15] I. Motedayen-Aval and A. Anastasopoulos, "Polynomial-complexity non-coherent symbol-by-symbol detection with application to adaptive iterative decoding of turbo-like codes," *IEEE Trans. Commun.*, vol. 51, no. 2, pp. 197–207, 2003.

[16] C. Morlet, M.-L. Boucheret, and I. Buret, "A carrier phase estimator for multi-media satellite payloads suited to RSC coding schemes," in *Proc. IEEE Int. Conf. Communications (ICC)*, 2000, pp. 455–459.

[17] C. Langlais and M. Helard, "Phase carrier for turbo codes over a satellite link with the help of tentative decisions," in *Proc. Int. Symp. Turbo Codes and Related Topics (ISTC)*, 2000, pp. 439–442.

[18] V. Lottici and M. Luise, "Embedding carrier phase recovery into iterative decoding of turbo-coded linear modulations," *IEEE Trans. Commun.*, vol. 52, no. 4, pp. 661–669, Apr. 2004.

[19] L. Zhang and A. Burr, "A novel carrier phase recovery method for turbo coded QPSK systems," presented at the Eur. Wireless (EW) Conf., Florence, Italy, 2002.

[20] S. L. Howard and C. Schlegel, "Differential turbo-coded modulation with APP channel estimation," *IEEE Trans. Commun.*, vol. 54, no. 8, pp. 1397–1406, Aug. 2006.

[21] M. Tuchler, R. Otnes, and A. Schmidbauer, "Performance of soft iterative channel estimation in turbo equalization," in *Proc. IEEE Int. Conf. Communications (ICC)*, 2002, pp. 1852–1858.

[22] N. Noels, V. Lottici, A. Dejonghe, H. Steendam, M. Moeneclaey, M. Luise, and L. Vandendorpe, "A theoretical framework for soft information based synchronization

in iterative (turbo) receivers," *EURASIP J. Wireless Commun. Netw.*, vol. 2005, no. 2, pp. 117–129, Apr. 2005.

[23] X. Wu and H. Xiang, "Iterative carrier phase recovery methods in turbo receivers," *IEEE Commun. Lett.*, vol. 9, no. 8, 2005.

[24] C. Herzet, X. Wautelet, V. Ramon, and L. Vandendorpe, "Iterative synchronization: EM algorithm versus Newton-Raphson approach," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2006, pp. 393–396.

[25] C. Herzet, V. Ramon, and L. Vandendorpe, "A theoretical framework for iterative synchronization based on the sum-product and the expectation-maximization algorithms," *IEEE Trans. Signal Process.* [Online]. Available: http://www.tele.ucl.ac.be/digicom/herzet/publications.php

[26] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[27] S. M. Aji and R. J. McEliece, "The generalized distributive law," *IEEE Trans. Inform. Theory*, vol. 46, no. 2, pp. 325–343, Mar. 2000.

[28] G. Colavolpe, A. Barbieri, and G. Caire, "Algorithms for iterative decoding in the presence of strong phase noise," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 9, pp. 1748–1757, Sep. 2005.

[29] J. M. Mendel, *Lessons in Estimation Theory for Signal Processing Communications and Control*, ser. Signal Processing. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[30] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[31] A. J. Viterbi and A. M. Viterbi, "Nonlinear estimation of PSK-modulated carrier phase with application to burst digital transmission," *IEEE Trans. Inf. Theory*, vol. IT-29, no. 4, pp. 543–551, Jul. 1983.

[32] M. Oerder and H. Meyr, "Digital filter and square timing recovery," *IEEE Trans. Commun.*, vol. 36, no. 5, pp. 605–611, May 1988.

[33] N. Noels, H. Wymeersch, 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. Commun.*, vol. 52, no. 3, pp. 473–483, Mar. 2004.

[34] N. Noels, H. Steendam, and M. Moeneclaey, "The true Cramer-Rao bound for carrier frequency estimation from a PSK signal," *IEEE Trans. Commun.*, vol. 52, no. 5, pp. 834–844, May 2004.

[35] A. D'Andrea, U. Mengali, and R. Reggiannini, "The modified Cramer-Rao bound and its application to synchronization problems," *IEEE Trans. Commun.*, vol. 42, no. 2/3/4, pp. 1391–1399, Feb./Mar./Apr. 1994.

[36] F. Gini, R. Regiannini, and U. Mengali, "The modified Cramer-Rao bound in vector parameter estimation," *IEEE Trans. Commun.*, vol. 64, no. 1, pp. 52–60, Jan. 1998.

[37] 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. Commun.*, vol. 46, no. 11, pp. 1536–1544, Nov. 1998.

[38] 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*, 2003, pp. 2933–2937.

[39] C. Herzet, "Code-Aided Synchronization for Digital Burst Communications," Ph.D. dissertation. [Online]. Available: http://www.tele.ucl.ac.be/digicom/herzet/index.php

[40] J. Dauwels, S. Korl, and H.-A. Loeliger, "Steepest descent on factor graphs," in *Proc. IEEE ITSOC Information Theory Workshop Coding and Complexity 2005*, pp. 42–46.

[41] C. Herzet, C. Ramon, and L. Vandendorpe, "Turbo-synchronization: A combined sum-product and expectation-maximization algorithm approach," in *Proc. IEEE Workshop Signal Processing Advances in Wireless Communications (SPAWC)*, 2005, pp. 191–195.

[42] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum-likelihood from incomplete data via the EM algorithm," *J. Roy. Stat. Soc.*, vol. 39, no. 1, pp. 1–38, Jan. 1977.

[43] C. F. J. Wu, "On the convergence properties of the EM algorithm," *Ann. Stat.*, vol. 11, no. 1, pp. 95–103, 1983.

[44] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. IT-20, no. 2, pp. 284–287, Mar. 1974.

[45] T. Heskes, O. Zoeter, and W. Wiegerinck, "Approximate expectation maximization," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. Saul, and B. Schölkopf, Eds. Cambridge, MA: MIT Press, 2004.

[46] C. Herzet, V. Ramon, L. Vandendorpe, and M. Moeneclaey, "EM algorithm-based timing synchronization in turbo receivers," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2003, pp. 612–615.

[47] V. Ramon, C. Herzet, L. Vandendorpe, and M. Moeneclaey, "EM algorithm-based multiuser synchronization in turbo receivers," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2004, pp. 849–852.

[48] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA: Athena Scientific, 2003.

[49] G. J. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*, ser. Probability and Statistics. New York: Wiley, 1997.

[50] A. P. Worthen and W. E. Stark, "Unified design of iterative receivers using factor graphs," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 843–849, Feb. 2001.

[51] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2005.

[52] C. Schlegel and A. Grant, "Differential space-time turbo codes," *IEEE Trans. Inf. Theory*, vol. 49, no. 9, pp. 2298–2306, Sep. 2003.

[53] T. M. Cassaro and C. N. Georghiades, "Frame synchronization for coded systems over AWGN channel," *IEEE Trans. Commun.*, vol. 52, no. 2, pp. 484–489, Mar. 2004.

[54] H. Wymeersch and M. Moeneclaey, "Iterative code-aided ML phase estimation and phase ambiguity resolution," *EURASIP J. Appl. Signal Process.*, vol. 2005, no. 6, 2005.

[55] C. Herzet, X. Wautelet, V. Ramon, and L. Vandendorpe, "Frame-error-rate-wise optimal code-aided hypothesis testing," in *Proc. IEEE Int. Conf. Communications (ICC)*, 2006, vol. 7, pp. 3162–3166.

ABOUT THE AUTHORS

Cédric Herzet was born in Verviers, Belgium in 1978. He received the electrical engineering degree and the Ph.D. degree in applied science from the Université Catholique de Louvain (UCL), Louvain-la-Neuve, Belgium, in 2001 and 2006, respectively. His graduation thesis concerned multiuser turbo receivers for WCDMA (Wideband Code Division Multiple Access) transmissions. His Ph.D. thesis dealt with code-aided synchronization for digital burst communications.

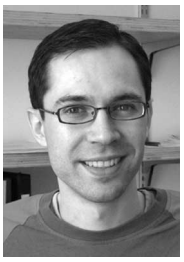
From August 2001 to April 2006, he was a Research Assistant in the Communications and Remote Sensing Laboratory of UCL in the digital communications group (DIGICOM). From August 2001 to December 2001, he worked on the topic of VDSL systems in collaboration with Alcatel Bell, Antwerp, Belgium. From January 2002 to April 2006, his research concerned synchronization and parameter estimation in iterative receivers. From May to August 2006, he was a Postdoctoral Researcher with the Ecole Normale Supérieure de Cachan, France. He is currently a Fulbright Postdoctoral Researcher at the University of California, Berkeley.

Nele Noels received the diploma of electrical engineering from Ghent University, Ghent, Belgium in 2001. She is currently working toward the Ph.D degree in the Department of Telecommunications and Information Processing, Ghent University. She is the author of several papers in international journals and conference proceedings. Her main research interests are in carrier and symbol synchronization.

Vincenzo Lottici received the Dr.Ing. degree (*cum laude*) in electronic engineering from the University of Pisa, Italy, in 1985.

From 1987 to 1993 he worked in the design and development of sonar digital signal processing systems. Since 1993 he has been with the Department of Information Engineering of the University of Pisa, where he is currently a Research Fellow and Assistant Professor in Telecommunications. His research interests include the area of wireless multicarrier and UWB systems, with particular emphasis on synchronization and channel estimation techniques.

Prof. Lottici received the Best Thesis SIP Award from the University of Pisa in 1986.



Henk Wymeersch is a postdoctoral associate with the Laboratory for Information and Decision Systems (LIDS) at the Massachusetts Institute of Technology (MIT). He obtained the Ph.D. degree in electrical engineering in 2005 from Ghent University, Belgium. In 2005–2006, Henk Wymeersch was a postdoctoral fellow of the Belgian American Educational Foundation at MIT, and in 2006 he won the Alcatel Bell Scientific Award for his Ph.D. thesis. He is a member of the IEEE, and author of the forthcoming book from Cambridge University Press "Iterative Receiver Design." His research interests include algorithm design for wireless transmission, statistical inference and iterative processing. A list of his publications can be found at http://www.mit.edu/hwymeers/.

Marco Luise (Senior Member, IEEE) received the M.Eng. and Ph.D. degrees in electronic engineering from the University of Pisa, Italy.

He was a Research Fellow of the European Space Agency (ESA) at ESTEC Noordwijk, The Netherlands, and a Researcher of the Italian National Research Council (CNR), at the CSMDR Pisa. He is currently a Full Professor of Telecommunications at the University of Pisa. He has authored more than 150 publications in international journals and contributions to major international conferences, and holds a few international patents. His main research interests lie in the area of wireless communications, with particular emphasis on CDMA/multicarrier signals and satellite communications and positioning.

Prof. Luise co-chaired four editions of the Tyrrhenian International Workshop on Digital Communications, was the General Chairman of the URSI Symposium ISSSE'98, and the General Chairman of EUSIPCO 2006 in Florence. He served as Editor for Synchronization of the IEEE TRANSACTIONS ON COMMUNICATIONS, and Editor for Communications Theory of the *European Transactions on Telecommunications*. He is the co-Editor-in-Chief of the recently founded *International Journal of Navigation and Observation*, and acts as General Secretary of the Italian Association GTTI, Gruppo Telecomunicazioni Teoria dell'Informazione.

Marc Moeneclaey (Fellow, IEEE) received the diploma of electrical engineering and the Ph.D. degree in electrical engineering from the University of Ghent, Ghent, Belgium, in 1978 and 1983, respectively.

From 1978 to 1999, he held at Ghent University various positions for the Belgian National Fund for Scientific Research (NFWO), from Research Assistant to Research Director. He is presently a Professor in the Department of Telecommunications and Information Processing (TELIN), Ghent University. His research interests include statistical communication theory, carrier and symbol synchronization, bandwidth- efficient modulation and coding, spread-spectrum, and satellite and mobile communication. He is the author of more than 200 scientific papers in international journals and conference proceedings. Together with Prof. H. Meyr (RWTH Aachen) and Dr. S. Fechtel (Siemens AG), he coauthored *Digital Communication Receivers—Synchronization, Channel Estimation, and Signal Processing* (New York: Wiley, 1998).

Prof. Moeneclaey served as Editor for Synchronization for the IEEE TRANSACTIONS ON COMMUNICATIONS from 1992 to 1994. He was co-guest editor for the Dec. 2001 IEEE JSAC special issue on Signal Synchronization in Digital Transmission Systems. From 1993 to 2002, he has been an executive Committee Member of the IEEE Communications and Vehicular Technology Society Joint Chapter, Benelux Section. He has been active in various international conferences as Technical Program Committee member and Session chairman.

Luc Vandendorpe (Fellow, IEEE) was born in Mouscron, Belgium, in 1962. He received the electrical engineering degree (*summa cum laude*) and the Ph. D. degree from the Université Catholique de Louvain (UCL) Louvain-la-Neuve, Belgium in 1985 and 1991, respectively.

Since 1985, he has been with the Communications and Remote Sensing Laboratory of UCL. In 1992, he was a Visiting Scientist and Research Fellow at the Telecommunications and Traffic Control Systems Group of the Delft Technical University, The Netherlands. Presently he is Full Professor and head of the Electrical Engineering Department of UCL. He is mainly interested in digital communication systems: equalization, joint detection/synchronization for CDMA, OFDM (multicarrier), MIMO and turbo-based communications systems (UMTS, xDSL, WLAN, etc.), and joint source/channel (de)coding.

Prof. Vandendorpe was corecipient of the Biennal Alcatel-Bell Award from the Belgian National Science Foundation (NSF) in 1990. In 2000 he was corecipient (with J. Louveaux and F. Deryck) of the Biennal Siemens Award from the Belgian NSF. He is or has been TPC member for IEEE VTC Fall 1999, IEEE Globecom 2003 Communications Theory Symposium, the 2003 Turbo Symposium, IEEE VTC Fall 2003, IEEE SPAWC 2005, and IEEE SPAWC 2006. He was co-technical chair (with P. Duhamel) for IEEE ICASSP 2006. He is an elected member of the Sensor Array and Multichannel Signal Processing committee of the Signal Processing Society and Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING.