

# A Theoretical Framework for Iterative Synchronization based on the Sum-Product and the Expectation-Maximization Algorithms

Cédric Herzet, Valéry Ramon and Luc Vandendorpe

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

Phone: +32 10 45 80 66, Fax: +32 10 45 20 89

{herzet, ramon, vandendorpe}@tele.ucl.ac.be

## Abstract

This paper deals with maximum-likelihood (ML) estimation of synchronization parameters for coded transmission systems. In particular we present a unified framework based on both the sum-product (SP) algorithm and the expectation-maximization (EM) algorithm for the design of iterative synchronizers. The proposed approach is shown to encompass some known iterative synchronizers. In particular, we revisit a previously-proposed framework based on the EM algorithm only by means of our "SP-EM" approach. The performance of the proposed synchronization method is assessed in terms of mean square error and bit-error rate by simulation results. In particular, we consider the joint synchronization of the timing epoch and the carrier phase offset in the case of convolutionally-coded and turbo-coded transmissions.

## INTRODUCTION

In digital communications the purpose of any receiver is to decrease as much as possible the number of wrong decisions made on the transmitted data. In that sense the optimal receiver is the one which minimizes the probability of making an error on the transmitted bits. Unfortunately, the practical implementation of such a receiver often turns out to be far too complex. This issue has led to an increasing interest in iterative algorithms approximating the optimal receiver. In particular, the so-called "turbo principle" discovered by Berrou and Glavieux [1] has been shown to exhibit excellent performance when applied to various receiver tasks: joint decoding of concatenated codes, joint demodulation and decoding [2], etc. Although obviously powerful, this principle has not yet been placed into a strong mathematical framework. Recently, graphical models have however been proposed to give an insight into

the convergence of these iterative algorithms. In particular, factor graphs [3] and the associated sum-product (SP) algorithm have been shown to enable a particularly clean derivation of iterative algorithms.

In addition to detection and decoding, a receiver has also to estimate a number of synchronization parameters in order to work properly. Most common synchronization parameters are the carrier phase offset, the carrier frequency offset, the timing epoch, etc. In a burst transmission system, most of the classical synchronizers are derived from the maximum-likelihood (ML) criterion and usually work in data-aided (DA) or non-data-aided (NDA) modes [4]. DA synchronizers lead to expenses in terms of bandwidth and power and should therefore be avoided in practice as much as possible. Conventional NDA estimators [4] are based on some approximations of the (log-)likelihood function and therefore do not compute the actual ML solution. Moreover, NDA estimators do not exploit any knowledge about the error correcting code which may be used for the transmission. For systems operating at low SNRs, these synchronizers may then fail to provide reliable enough estimates of the synchronization parameters and lead therefore to bit-error-rate (BER) degradation.

An idea proposed by several authors in order to deal with the problems related to the NDA nature of conventional estimators is to implement iterative synchronizers which take benefit from the soft information available at the decoder output in iterative receivers. Such synchronizers are then expected to get closer to the performance of a true ML synchronizer exploiting the code knowledge. This approach is often referred to as *turbo-synchronization*. Most of the turbo-synchronizers available in the current technical literature do not derive from a theoretical framework enabling a systematic derivation of the synchronization algorithms. We mention [5], [6], [7] and [8] which are ML-based turbo-synchronization algorithms, i.e. the proposed synchronizers maximize a "modified" log-likelihood function which is built by using the soft information available in the iterative receiver. In particular, papers [5]-[7] basically follow the same approach and propose to use the extrinsic probabilities delivered by an iterative receiver as symbol a priori probabilities in the log-likelihood function. Contribution [5] deals with timing estimation whereas [6] and [7] consider carrier phase recovery respectively in the case of a turbo-coded transmission and a bit-interleaved coded modulation (BICM) transmission. In [8] the log-likelihood function is expanded into a first-order polynomial and the symbol a priori probabilities are approximated by the (so-called) a posteriori probabilities delivered by the soft-in soft-out (SISO) modules which make up the iterative receiver.

Among papers which try to give a systematic mathematical approach to turbo synchronization we may mention [9] which proposes a framework based on the expectation-maximization (EM) algorithm [10]. In particular, the latter framework provides a justification to the phase synchronizer proposed in [8]. The EM-based approach presented in [9] has been applied in several contributions, e.g. [11], [12]. More recently, the good performance obtained by iterative receivers based on the factor-graph representation and the sum-product (SP) algorithm has led some authors to place turbo synchronization into this framework. In particular, in [13] a message-passing phase estimator is designed by using an approach based on density evolution. In [14], the authors deal with the design of iterative receivers in the presence of a carrier phase uncertainty. The proposed approach relies on some approximations of the messages computed by the SP algorithm.

In this paper we place the general issue of the synchronization of a linearly data-modulated bandpass signal, i.e. the estimation of the carrier phase offset, the carrier frequency offset, the timing epoch and the real gain, into the factor-graph framework. We consider the case of constant parameters over the block bursts. Relevant references dealing with time-varying parameter estimation may be found in the literature, see e.g. [15] and references therein. After having derived the general SP-message update equations, we show that several previously-proposed iterative synchronizers [5]-[9] may be understood in the context of the factor-graph representation. In particular, we make a connection between the EM-based synchronization framework proposed in [9] and the SP-based synchronization framework considered in this paper by deriving a synchronization method based on both the SP and the EM algorithms. The performance of the proposed synchronizer is compared through simulation results to the performance of some classical synchronization methods.

The remainder of the paper is organized as follows. In Section I, we describe the model and the notations used in the paper. In section II we relate the optimal receiver to the ML synchronization problem considered in this paper. A general formulation of the ML estimation problem as well as some conventional approaches to deal with its resolution are then given in section III. In section IV, we derive the general SP-algorithm equations for the design of iterative receivers in the presence of synchronization parameter uncertainties. Then, combining both the EM and the SP algorithms in section V, we show in section VI that the EM-based framework proposed in [9] may be viewed as a particular case of a more general SP-EM framework. Finally in section VII, we assess the performance of the proposed SP-EM synchronization method in the case of the joint carrier phase and timing synchronization.

## I. SYSTEM MODEL

We consider a burst transmission system where a sequence of  $L$  information bits  $\mathbf{u}$  is encoded by a channel encoder with code rate  $R$ . The resulting coded-bit sequence  $\mathbf{x}$  is mapped onto a complex signaling constellation  $\mathcal{A}$  of size  $|\mathcal{A}|$  and shaped by a unit-energy square-root raised-cosine pulse  $g(t)$  with roll-off  $\alpha$ . The resulting baseband signal is

$$s(t) = \sum_{k=0}^{K-1} a_k g(t - kT), \quad (1)$$

where  $a_k \in \mathcal{A}$  are the transmitted symbols,  $T$  is the symbol duration and  $K$  is the number of symbols in the burst. After propagation through an additive-white-Gaussian-noise (AWGN) channel, the received signal can be written as

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

where  $w(t)$  is the complex envelope of an AWGN with passband two-sided power spectral density  $N_0/2$ ,  $A$  denotes the channel gain,  $\tau$  the symbol timing offset,  $\nu$  and  $\theta$  respectively the carrier-frequency and the carrier-phase offsets. The carrier frequency offset is assumed to be upper-bounded by  $\nu_{max}$ , i.e.  $-\nu_{max} \leq \nu \leq \nu_{max}$ .

In the receiver, after anti-aliasing filtering, signal  $r(t)$  is sampled at a rate of  $1/T_s$  (with  $1/T_s > 2\nu_{max} + (1 + \alpha)/T$ ) leading to samples

$$r(lT_s) = \sum_{k=0}^{K-1} a_k g(lT_s - kT - \tau) e^{j(2\pi\nu lT_s + \theta)} + w(lT_s). \quad (3)$$

Since the sampling rate satisfies Nyquist's criterion [16], samples  $r(lT_s)$  contains the same information as the continuous-time signal  $r(t)$  and may therefore be used to estimate the unknown parameters, namely the transmitted symbols and the synchronization parameters, underlying the received signal.

## II. OPTIMAL RECEIVER AND ML SYNCHRONIZATION

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

Let  $\mathbf{r}$  denote the vector obtained by stacking observed samples  $r(lT_s)$ . The symbol-wise optimal receiver is the one which enables to minimize the symbol error probability or equivalently,

$$\hat{a}_k = \arg \max_{a_k} p(a_k|\mathbf{r}), \quad (4)$$

where  $\hat{a}_k$  is the decision made on transmitted symbol  $a_k$ . Let  $\mathbf{b} \triangleq [A, \tau, \nu, \theta]^T$  indicate the vector of synchronization parameters. Probability  $p(a_k|\mathbf{r})$  may be rewritten as the marginal of a joint probability:

$$p(a_k|\mathbf{r}) = \int_{\mathbf{b}} p(a_k|\mathbf{r}, \mathbf{b}) p(\mathbf{b}|\mathbf{r}) d\mathbf{b}. \quad (5)$$

Unfortunately, due to the integration over the unknown synchronization parameters, the computation of  $p(a_k|\mathbf{r})$  in (5) is usually intractable and one has therefore to resort to approximations. A common approach consists in approximating  $p(\mathbf{b}|\mathbf{r})$  as

$$p(\mathbf{b}|\mathbf{r}) \simeq \delta(\mathbf{b} - \hat{\mathbf{b}}), \quad (6)$$

i.e., one assumes that probability density  $p(\mathbf{b}|\mathbf{r})$  is concentrated around one point  $\hat{\mathbf{b}}$ . Of course, more-elaborated approximations can be considered. However, it may be shown [17] that approximation (6) is quite accurate as long as the number of available observations is large as compared to the dimension of vector  $\mathbf{b}$ .

Using approximation (6), the overall receiver complexity remains very close to the complexity of a receiver which would perfectly know synchronization parameter values. Indeed, replacing, in (5),  $p(\mathbf{r}|\mathbf{b})$  by its approximation (6) we come up with

$$p(a_k|\mathbf{r}) = p(a_k|\mathbf{r}, \hat{\mathbf{b}}). \quad (7)$$

It is then clear from (7) that the only extra complexity required by this approach is the computation of a proper point  $\hat{\mathbf{b}}$ . This task is referred to as synchronization. Note that a priori statistical information about  $\mathbf{b}$  is usually assumed to be uniform, i.e.  $p(\mathbf{b}) = C^{st}$  on the domain of  $\mathbf{b}$ . In that case we have that

$$p(\mathbf{b}|\mathbf{r}) \sim p(\mathbf{r}|\mathbf{b}), \quad (8)$$

where  $\sim$  denotes equality up to a normalization factor. We may therefore equivalently search a proper point  $\hat{\mathbf{b}}$  of  $p(\mathbf{r}|\mathbf{b})$ . Note, to avoid confusion, that the received observation vector  $\mathbf{r}$  is known and fixed. Therefore, in the sequel,  $p(\mathbf{r}|\mathbf{b})$  is regarded as a function of  $\mathbf{b}$ . Several options are possible for the choice of  $\hat{\mathbf{b}}$ , see e.g. [17]. In practice, since it exhibits very good asymptotical properties, most of the commonly-used synchronizers are based on the maximum-likelihood (ML) criterion. We give a general formulation of this criterion in the next section.

### III. MAXIMUM-LIKELIHOOD PARAMETER ESTIMATION

Let  $\mathbf{a}$  denote the vector of transmitted symbols i.e.  $\mathbf{a} = [a_0, a_1, \dots, a_{K-1}]^T$ . The problem addressed in the sequel of this paper is to find the ML estimate  $\hat{\mathbf{b}}_{\text{ML}}$  of  $\mathbf{b}$ , that is to say the solution of

$$\hat{\mathbf{b}}_{\text{ML}} = \arg \max_{\mathbf{b}} \left\{ \ln p(\mathbf{r}|\mathbf{b}) \right\}, \quad (9)$$

where

$$p(\mathbf{r}|\mathbf{b}) = \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a}, \mathbf{b}) p(\mathbf{a}). \quad (10)$$

Probability  $p(\mathbf{a})$  represents the a priori knowledge we have about the transmitted sequence. Of course, this a priori information depends on the kind of transmission we are considering. For example, in the case of uncoded transmission the data symbols are independent and equally distributed i.e.

$$p(\mathbf{a}) = |\mathcal{A}|^{-K} \quad \forall \mathbf{a} \in \mathcal{A}^K, \quad (11)$$

whereas for a coded transmission there is only a subset  $\mathcal{B} \subset \mathcal{A}^K$  of all possible sequences which corresponds to the legitimate encoder output sequences. In this case, the a priori distribution  $p(\mathbf{a})$  may be written as

$$p(\mathbf{a}) = \begin{cases} |\mathcal{A}|^{-KR} & \forall \mathbf{a} \in \mathcal{B} \\ 0 & \forall \mathbf{a} \notin \mathcal{B}. \end{cases} \quad (12)$$

Conventionally, transmissions are coded and we are therefore required to find the solution of (9) given (12). Unfortunately, the huge number of terms in (10) usually makes the direct computation of the "code-aided" (CA) ML estimate intractable. Consequently, rather than computing the exact CA ML estimate conventional synchronizers proposed in the literature [4] are smart approximations of the true ML solution. These conventional synchronizers may basically be divided into data-aided (DA) and non-data-aided (NDA) synchronizers. More recently, a synchronizer based on the EM algorithm [10] has been proposed in [9] to iteratively compute the exact CA ML solution (9). These different approaches are developed in more details in the remainder of this section.

#### *Data-aided conventional synchronizers*

A common approach to estimate synchronization parameters consists in transmitting a sequence of  $K_{\text{pil}}$  known symbols (pilots), say  $\mathbf{a}_{\text{pil}}$ , and applying the ML criterion on the observation subset  $\mathbf{r}_{\text{pil}}$  which only depends on the pilot symbols i.e.

$$\hat{\mathbf{b}} = \arg \max_{\mathbf{b}} \left\{ \ln p(\mathbf{r}_{\text{pil}}|\mathbf{b}) \right\}. \quad (13)$$

In this case, since the transmitted sequence is a priori known the likelihood function  $p(\mathbf{r}_{\text{pil}}|\mathbf{b})$  may be computed very easily and the corresponding maximization problem becomes therefore tractable. Note however that the parameter estimate is computed by maximizing  $p(\mathbf{r}_{\text{pil}}|\mathbf{b})$  rather than the actual goal function  $p(\mathbf{r}|\mathbf{b})$ . This approach is therefore clearly suboptimal since the part of the observations related to the data symbols is not taken into account in the computation of the estimate. As a consequence an increase of the estimation quality also implies an increase of either the number of transmitted pilots or the power allocated to the pilot sequence. Both solutions lead to a waste of resources and in practice this costly approach should be avoided as most as possible.

#### *Non-data-aided conventional synchronizers*

As mentioned above, DA synchronizers do not exploit the whole observation set and lose therefore a part of the available information. Conventional NDA synchronizers provide a solution to this problem by making a different assumption on the transmitted sequence: although the transmission may be coded, NDA synchronizers assume that the transmitted sequence is a priori distributed according to (11) i.e. all possible sequences are a priori equiprobable. Assuming a low-SNR operating point, approximations [4] enable to derive closed-form expressions for the estimation of synchronization parameters. Well-known synchronizers such as the Viterbi&Viterbi [18] phase estimator and the Oerder&Meyr [19] timing estimator are based on this approach. These synchronizers exhibit a low complexity and are therefore well-suited for practical implementation. They are however not optimal since they are based on both a low SNR approximation and an uncoded-transmission assumption.

#### *Code-aided Synchronizer based on the EM Algorithm*

In some systems operating at low SNRs the suboptimality of conventional (DA and NDA) synchronizers may prevent them from providing parameter estimates which are reliable enough. This may in turn lead to important bit-error-rate (BER) degradation. Consequently, state-of-the-art receivers, which operate at very low SNRs, require methods to accurately estimate synchronization parameters. Recently, an iterative code-aided synchronizer based on the EM algorithm has been proposed [9]. This synchronizer is briefly presented in this subsection.

The expectation-maximization (EM) algorithm, first defined by Dempster, Laird and Rubin in [10], is an iterative method which falls into the framework of ML estimation. This method is well-suited to

situations where the ML estimation of a given parameter  $\mathbf{b}$  would be straightforward if some additional data were known. For example, in the problem we are dealing with the solution of maximization problem (9) would be straightforward if the knowledge of the transmitted sequence  $\mathbf{a}$  was available. In such cases the observed data vector  $\mathbf{r}$  may be viewed as being *incomplete* and regarded as an observable function of an extended data set  $\mathbf{z}$  referred to as *complete data set*. Using this formalism the EM algorithm states that the sequence  $\{\hat{\mathbf{b}}^{(n)}\}_{n=0}^{\infty}$  defined by

$$Q(\mathbf{b}, \hat{\mathbf{b}}^{(n)}) = \int_{\mathbf{z}} p(\mathbf{z}|\mathbf{r}, \hat{\mathbf{b}}^{(n)}) \ln p(\mathbf{z}|\mathbf{b}) d\mathbf{z} \quad (14)$$

$$\hat{\mathbf{b}}^{(n+1)} = \arg \max_{\mathbf{b}} Q(\mathbf{b}, \hat{\mathbf{b}}^{(n)}), \quad (15)$$

converges under some general conditions [20] to the ML estimate (9).

The nice convergence properties of the EM algorithm have led a number of authors to apply it to parameter estimation issues in digital communication. In particular, in the case of linearly-modulated bandpass signal synchronization, it has been shown in [9] that the EM update equations (14), (15) become:

$$[\hat{\nu}^{(n)}, \hat{\tau}^{(n)}] = \arg \max_{\nu, \tau} \left\{ \left| \sum_{k=0}^{K-1} \eta_k^*(\mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) y_k(\nu, \tau) \right| \right\} \quad (16)$$

$$\hat{\theta}^{(n)} = \arg \left\{ \sum_{k=0}^{K-1} \eta_k^*(\mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) y_k(\hat{\nu}^{(n)}, \hat{\tau}^{(n)}) \right\} \quad (17)$$

$$\hat{A}^{(n)} = \frac{|\sum_{k=0}^{K-1} \eta_k^*(\mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) y_k(\hat{\nu}^{(n)}, \hat{\tau}^{(n)})|}{\sum_{k=0}^{K-1} \rho_k(\mathbf{r}, \hat{\mathbf{b}}^{(n-1)})}, \quad (18)$$

where

$$y_k(\nu, \tau) \triangleq \int_{-\infty}^{+\infty} r(t) e^{-j(2\pi\nu t)} g(t - kT - \tau) dt, \quad (19)$$

$$= T_s \sum_l r(lT_s) e^{-j(2\pi\nu lT_s)} g(lT_s - kT - \tau), \quad (20)$$

i.e.  $y_k(\nu, \tau)$  is the matched filter output computed at time  $kT + \tau$  assuming a frequency offset  $\nu$ . Notations  $\eta_k(\mathbf{r}, \hat{\mathbf{b}}^{(n-1)})$  and  $\rho_k(\mathbf{r}, \hat{\mathbf{b}}^{(n-1)})$  denote the first and second order moments of symbol  $a_k$  given current estimate  $\hat{\mathbf{b}}^{(n-1)}$  and observation vector  $\mathbf{r}$  i.e.

$$\eta_k(\mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) \triangleq \sum_{a \in \mathcal{A}} a p(a_k = a | \mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) \quad (21)$$

$$\rho_k(\mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) \triangleq \sum_{a \in \mathcal{A}} |a|^2 p(a_k = a | \mathbf{r}, \hat{\mathbf{b}}^{(n-1)}). \quad (22)$$

As mentioned above, the sequence  $\{\hat{\mathbf{b}}^{(n)}\}_{n=0}^{\infty}$  will converge (under some conditions) to the solution of the ML problem (9) given (12). In other words, the solution computed by the EM-based synchronizer is code-aware, i.e., takes benefit from the sequence a priori information related to the code structure. Note however from (21) and (22) that the implementation of an EM-based iterative synchronizer requires the evaluation of marginal a posteriori probabilities  $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)})$ . In the case of a convolutionally-coded transmission, these a posteriori probabilities may be efficiently computed by means of a BCJR algorithm [21]. However, in the general case, the computation of these a posteriori probabilities is unfortunately intractable. A common approach [9] in iterative receivers consists therefore in using as a posteriori probabilities, the probabilities, say  $\tilde{p}(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)})$ , delivered at the output of the constituent soft-in soft-out (SISO) modules. Note that since these probabilities are not equal to the true a posteriori probabilities, the nice convergence properties of the EM algorithm may no longer be ensured. In section VI, we will give a justification to this approach by showing that the EM-based mathematical framework proposed in [9] may actually be viewed as an approximation of a more general framework based on both the SP and the EM algorithms.

#### IV. CODE-AIDED SYNCHRONIZATION BASED ON THE FACTOR GRAPHS AND THE SUM-PRODUCT ALGORITHM

In this section, after a short introduction to the factor graph representation and the associated SP algorithm, we place the issue of the receiver design in the presence of synchronization parameter uncertainties into the factor-graph framework.

##### *Factor Graphs and the Sum-Product Algorithm*

Let  $v_1, v_2, \dots, v_n$  denote a collection of variables and let  $g(v_1, v_2, \dots, v_n)$  denote a global function which may be factorized as

$$g(v_1, v_2, \dots, v_n) = \prod_j f_j(v_{Q_j}) \quad (23)$$

where  $v_{Q_j}$  are sets of elements from  $\{v_1, v_2, \dots, v_n\}$ . In many applications we are interested in computing efficiently the marginal functions of  $g(v_1, v_2, \dots, v_n)$  i.e.

$$g(v_i) = \sum_{\sim\{v_i\}} g(v_1, v_2, \dots, v_n) \quad (24)$$

where the notation  $\sim\{v_i\}$  denotes the summation over all the variables but  $v_i$ . Factor graph representation and the associated sum-product (SP) algorithm [3] provide a general framework to efficiently solve this

problem.

A factor graph is a bipartite graph which expresses the structure of the factorization (23). 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 an efficient procedure which enables to compute (either exactly or approximately) the marginals of a global function by passing messages along the edges of the corresponding factor graph. Denoting by  $\mu_{v_i \rightarrow f_j}(v_i)$  the message sent from node  $v_i$  to node  $f_j$  and by  $\mu_{f_j \rightarrow v_i}(v_i)$  the message sent from node  $f_j$  to node  $v_i$ , the message computations performed by the SP-algorithm may be expressed as follows:

$$\mu_{v_i \rightarrow f_j}(v_i) = \prod_{h \in n(v_i) \setminus \{f_j\}} \mu_{h \rightarrow v_i}(v_i) \quad (25)$$

$$\mu_{f_j \rightarrow v_i}(v_i) = \sum_{\sim\{v_i\}} \left( f_j(v_{Q_j}) \prod_{y \in n(f_j) \setminus \{v_i\}} \mu_{y \rightarrow f_j}(y) \right) \quad (26)$$

where  $n(q)$  denote the set of neighbors of a given node  $q$  in the factor graph. In [3], it is shown that if the factor graph is finite and cycle-free, the SP algorithm can compute in a finite number of steps the exact marginals of the function that the graph represents. These marginals are equal to the product of the messages entering each variable node. 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 may or may not perform well, depending on the chosen code structure, block length, SNR, etc.

#### *Suboptimal Iterative Receiver based on the SP algorithm*

As mentioned above, the sum-product (SP) algorithm is a message-passing algorithm which operates on factor graphs and enables to exactly or approximately compute marginals of the function that the graph represents. In this section we show that the factor-graph framework and the SP algorithm may be used to design iterative suboptimal receiver in the presence of unknown synchronization parameters.

Let us first observe that both the objective functions of maximization problems (4) and (9) may be written as

$$p(a_k | \mathbf{r}) \sim \sum_{\sim\{a_k\}} p(\mathbf{r}, \mathbf{a}, \mathbf{b})$$

$$p(\mathbf{r} | \mathbf{b}) \sim \sum_{\sim\{\mathbf{b}\}} p(\mathbf{r}, \mathbf{a}, \mathbf{b})$$

i.e., it may be regarded as the marginal of a global function  $p(\mathbf{r}, \mathbf{a}, \mathbf{b})$ . Marginal probabilities  $p(a_k|\mathbf{r})$  and  $p(\mathbf{r}|\mathbf{b})$ , or rather an approximation of them<sup>1</sup>, may therefore be computed by applying the SP algorithm to the factor graph associated with probability  $p(\mathbf{r}, \mathbf{a}, \mathbf{b})$ . This factor graph is represented in Fig. 11 by noticing that

$$p(\mathbf{r}, \mathbf{a}, \mathbf{b}) \sim p(\mathbf{r}|\mathbf{a}, \mathbf{b}) p(\mathbf{a}).$$

Note that probability  $p(\mathbf{a})$  may be factorized according to the code and the mapping which are considered for the transmission (see Appendix A for more details). However, for the sake of generality we do not explicitly represent the factor graph associated with  $p(\mathbf{a})$  in Fig. 11.

If a coded transmission is considered, the dependence between coded bits introduces cycles in the graph represented in Fig. 11. As mentioned in the previous section, a consequence of the presence of cycles in the factor graph is that the application of the SP algorithm leads to an iterative algorithm with no natural termination. It is therefore required to define a message-passing schedule which specifies in which order the messages are updated. Denoting by  $\mu_{a_k \rightarrow p}^{(m)}$  (resp.  $\mu_{p \rightarrow a_k}^{(m)}$ ) the message passing from variable node  $a_k$  (resp. factor node  $p(\mathbf{r}|\mathbf{a}, \mathbf{b})$ ) to factor node  $p(\mathbf{r}|\mathbf{a}, \mathbf{b})$  (resp. variable node  $a_k$ ) at iteration  $m$ , we define the following message passing schedule: at each iteration messages  $\mu_{a_k \rightarrow p}^{(m)}$  are first updated by applying the SP algorithm on the lower part of the factor graph in Fig. 1, i.e., by exploiting the code structure underlying the transmitted symbols sequence  $\mathbf{a}$ ; new messages  $\mu_{p \rightarrow a_k}^{(m+1)}$  are then computed by taking into account updated messages  $\mu_{a_k \rightarrow p}^{(m)}$ .

Using SP algorithm update rules, messages  $\mu_{p \rightarrow a_k}^{(m+1)}$  may be expressed as follows:

$$\mu_{p \rightarrow a_k}^{(m+1)}(a_k) \sim \sum_{\sim\{a_k\}} p(\mathbf{r}|\mathbf{a}, \mathbf{b}) \prod_{l \neq k} \mu_{a_l \rightarrow p}^{(m)}(a_l). \quad (27)$$

Denoting by  $\mu_{p \rightarrow \mathbf{b}}^{(m+1)}$  the message entering synchronization node  $\mathbf{b}$ , we also have

$$\mu_{p \rightarrow \mathbf{b}}^{(m+1)}(\mathbf{b}) \sim \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{a}, \mathbf{b}) \prod_l \mu_{a_l \rightarrow p}^{(m)}(a_l). \quad (28)$$

As mentioned above, if the graph contains cycles, the products of the messages entering the variable nodes are an approximation of the marginals of the function that the graph represents. Therefore, message  $\mu_{p \rightarrow \mathbf{b}}^{(m+1)}$  (resp.  $\mu_{p \rightarrow a_k}^{(m+1)}$ ,  $\mu_{a_k \rightarrow p}^{(m+1)}$ ) will give, at each SP iteration, an updated approximation of actual marginal  $p(\mathbf{r}|\mathbf{b})$  (resp.  $p(a_k|\mathbf{r})$ ). Maximization problem (9) (resp. (4)) may then be approximated by maximizing SP messages  $\mu_{p \rightarrow \mathbf{b}}^{(m+1)}$  (resp.  $\mu_{p \rightarrow a_k}^{(m+1)}$ ,  $\mu_{a_k \rightarrow p}^{(m+1)}$ ) rather than the actual objective function  $p(\mathbf{r}|\mathbf{b})$  (resp.  $p(a_k|\mathbf{r})$ ).

<sup>1</sup>because the factor graph of  $p(\mathbf{r}, \mathbf{a}, \mathbf{b})$  usually contains cycles.

*SP-Based Synchronizers: A Simplification of the SP Algorithm*

The implementation of the SP-algorithm update equations is too complex. Indeed, using Bayes rule, let us rewrite message  $\mu_{p \rightarrow a_k}^{(m+1)}$  as

$$\mu_{p \rightarrow a_k}^{(m+1)}(a_k) \sim \int_{\mathbf{b}} p_k^{(m)}(a_k | \mathbf{r}, \mathbf{b}) p_k^{(m)}(\mathbf{r} | \mathbf{b}) d\mathbf{b}, \quad (29)$$

where

$$p_k^{(m)}(a_k | \mathbf{r}, \mathbf{b}) \triangleq \frac{\sum_{\mathbf{a} \setminus \{a_k\}} p(\mathbf{r} | \mathbf{a}, \mathbf{b}) \prod_{l \neq k} \mu_{a_l \rightarrow p}^{(m)}(a_l)}{\sum_{\mathbf{a}} p(\mathbf{r} | \mathbf{a}, \mathbf{b}) \prod_{l \neq k} \mu_{a_l \rightarrow p}^{(m)}(a_l)} \quad (30)$$

$$p_k^{(m)}(\mathbf{r} | \mathbf{b}) \triangleq \sum_{\mathbf{a}} p(\mathbf{r} | \mathbf{a}, \mathbf{b}) \prod_{l \neq k} \mu_{a_l \rightarrow p}^{(m)}(a_l). \quad (31)$$

From (29) and (31) we may first notice that a different message  $p_k^{(m)}(\mathbf{r} | \mathbf{b})$  has to be computed for each  $k$ . Moreover, the integration operation required in (29) does usually not have an analytical solution.

A first approximation on the SP messages consists therefore in approximating message (31) by

$$\begin{aligned} p_k^{(m)}(\mathbf{r} | \mathbf{b}) &\simeq \sum_{\mathbf{a}} p(\mathbf{r} | \mathbf{a}, \mathbf{b}) \prod_l \mu_{a_l \rightarrow p}^{(m)}(a_l), \\ &= \mu_{p \rightarrow \mathbf{b}}^{(m+1)}(\mathbf{b}), \end{aligned} \quad (32)$$

i.e. in taking into account *all* the messages arriving at node  $p(\mathbf{r} | \mathbf{a}, \mathbf{b})$ . This approximation is motivated by the fact that, since the number of messages arriving at factor node  $p(\mathbf{r} | \mathbf{a}, \mathbf{b})$  is typically large, the effect on  $p_k^{(m)}(\mathbf{r} | \mathbf{b})$  of an additional factor  $\mu_{a_k \rightarrow p}^{(m)}(a_k)$  will almost be negligible. Making this approximation, we see that probability  $p_k^{(m)}(\mathbf{r} | \mathbf{b})$  no longer depends on index  $k$  and has therefore to be computed only once. Plugging (32) into (29), we then get

$$\mu_{p \rightarrow a_k}^{(m+1)}(a_k) \sim \int_{\mathbf{b}} p_k^{(m)}(a_k | \mathbf{r}, \mathbf{b}) \mu_{p \rightarrow \mathbf{b}}^{(m+1)}(\mathbf{b}) d\mathbf{b}. \quad (33)$$

Note the similarity between (5) and (33). The latter is equal to the former in which probabilities have been computed by considering the messages entering factor node  $p(\mathbf{r} | \mathbf{a}, \mathbf{b})$ , i.e.  $\mu_{a_k \rightarrow p}^{(m)}$ , as symbol a priori probability.

As mentioned above, the integral over the synchronization parameters in (33) does not have any simple analytical solution. In practice, we have therefore to resort to some approximations on message  $\mu_{p \rightarrow \mathbf{b}}^{(m)}$ . In particular,  $\mu_{p \rightarrow \mathbf{b}}^{(m)}$  may be approximated by a sample list i.e. by a function  $\tilde{\mu}_{p \rightarrow \mathbf{b}}^{(m)}$  defined as

$$\tilde{\mu}_{p \rightarrow \mathbf{b}}^{(m)}(\mathbf{b}) \sim \sum_i \mu_{p \rightarrow \mathbf{b}}^{(m)}(\hat{\mathbf{b}}_i^{(m)}) \delta(\mathbf{b} - \hat{\mathbf{b}}_i^{(m)}), \quad (34)$$

where  $\hat{\mathbf{b}}_i^{(m)}$  denotes the  $i^{\text{th}}$  sampling point of  $\mu_{p \rightarrow \mathbf{b}}^{(m)}$ . Note that approximation is usually accurate [17] if the frame length is large enough with respect to the dimensionality of  $\mathbf{b}$ . Doing this approximation, the integral in (33) reduces to a finite sum. Sampling points  $\hat{\mathbf{b}}_i^{(m)}$  may be computed in a variety of ways leading to as many different algorithms. For example in the context of carrier phase synchronization, the authors of [14] consider both the cases of equally-spaced sampling points and the case of sampling points computed thanks to a particle filter method.

In this paper, we will assume that the ambiguity problems related to synchronization parameter estimation, i.e., e.g. the frame synchronization or the phase-ambiguity resolution, are perfectly solved. State-of-the-art algorithms to deal with these problems may be found in [22], [23]. The resolution of these problems actually reduces the possible range for the synchronization parameter values. For example, if the frame synchronization problem is perfectly solved, we are ensured that  $-0.5 \leq \tau < 0.5$ . On this limited range, message  $\mu_{p \rightarrow \mathbf{b}}^{(m)}$  has only one maximum and may be well-approximated by

$$\tilde{\mu}_{p \rightarrow \mathbf{b}}^{(m)}(\mathbf{b}) = \delta(\mathbf{b} - \hat{\mathbf{b}}^{(m)}), \quad (35)$$

as long as the frame length is large enough with respect to the dimensionality of  $\mathbf{b}$  [17]. Note that approximation (35) is consistent in the context of synchronization since it is the same kind of approximation, i.e. (6), which justifies the synchronization operation in section II.

Using approximation (35), we have to compute at each SP iteration a representative point  $\hat{\mathbf{b}}^{(m)}$  of message  $\mu_{p \rightarrow \mathbf{b}}^{(m)}(\mathbf{b})$  according to some criteria. A common criterion consists in considering the maximum of the distribution i.e.

$$\hat{\mathbf{b}}^{(m)} = \arg \max_{\mathbf{b}} \mu_{p \rightarrow \mathbf{b}}^{(m)}(\tilde{\mathbf{b}}). \quad (36)$$

As long as the approximation  $\mu_{p \rightarrow \mathbf{b}}^{(m+1)}(\mathbf{b})$  of  $p(\mathbf{r}|\mathbf{b})$  improves at each SP iteration, we may expect the sequence of estimates  $\hat{\mathbf{b}}^{(m)}$  to get closer and closer to the ML solution (9). Note however that, although the efficiency of the SP algorithm has already been shown in a number of scenarios, its convergence to the actual marginal probabilities is not ensured when the considered factor graph has cycles. As a consequence, no theoretical conclusions may be drawn about the convergence of this approach. Instead, the powerfulness of this approach will be shown through simulation results in section VII.

## V. SYNCHRONIZATION BASED ON A COMBINATION OF THE SP AND THE EM ALGORITHMS

In the previous section, we show that iterative synchronizers may be designed by using the factor-graph representation and the SP algorithm. In this section, we emphasize that several synchronizers [5]-[9]

previously proposed in the literature may be understood in the factor-graph framework. In particular, we show that the EM-based synchronization framework proposed in [9] may be viewed as an approximation of a more general framework based on both the SP and the EM algorithm.

Let us first observe from (36) that the computation of the new synchronization parameter estimate  $\hat{\mathbf{b}}^{(m)}$  requires to solve a maximization problem at each SP iteration. Unfortunately this maximization problem can usually not be solved explicitly. Therefore, the solution of (36) has to be computed by means of iterative numerical methods. Since the SP algorithm is itself iterative, the overall synchronization algorithm then becomes doubly iterative.

In [14] for example, the authors solve maximization problem (36) in the context of carrier phase synchronization by using a gradient method i.e.

$$\hat{\mathbf{b}}^{(m,n)} = \hat{\mathbf{b}}^{(m,n-1)} + \lambda \left( \frac{\partial \mu_{p \rightarrow \mathbf{b}}^{(m)}(\tilde{\mathbf{b}})}{\partial \tilde{\mathbf{b}}} \right) \Big|_{\tilde{\mathbf{b}} = \hat{\mathbf{b}}^{(m,n-1)}}, \quad (37)$$

where  $\hat{\mathbf{b}}^{(m,n)}$  denotes the  $n^{\text{th}}$  estimate generated by the gradient method at the  $m^{\text{th}}$  SP algorithm iteration and  $\lambda$  is a step parameter which influences the convergence of the method: too small  $\lambda$  may slow down the convergence whereas too large  $\lambda$  leads to divergence. Note that the implementation of a gradient method requires the computation of message derivatives. Moreover, the convergence of the method requires the determination of a proper step parameter  $\lambda$ , which depends on both the SNR and the considered SP iteration.

In this paper, we propose to solve the intermediate maximization problem (36) via the expectation-maximization algorithm. As observed from section III, the EM maximization method requires neither the computation of the message derivative nor step-parameter tuning.

Let us first emphasize that message  $\mu_{p \rightarrow \mathbf{b}}^{(m)}$  may be regarded as a likelihood function. Indeed, considering (28) we see that this message may also be rewritten as

$$\mu_{p \rightarrow \mathbf{b}}^{(m+1)}(\mathbf{b}) = \sum_{\mathbf{a}} p(\mathbf{r}|\mathbf{b}, \mathbf{a}) p^{(m)}(\mathbf{a}), \quad (38)$$

where  $p^{(m)}(\mathbf{a}) \triangleq \prod_l \mu_{a_l \rightarrow p}^{(m)}(a_l)$ . Comparing (38) and (10), we may then notice that message  $\mu_{p \rightarrow \mathbf{b}}^{(m)}$  has exactly the same structure as the actual likelihood function (10). However, instead of considering the actual symbol a priori information  $p(\mathbf{a})$ , a modified a priori information  $p^{(m)}(\mathbf{a})$  is considered in (38).

As mentioned in section III, maximum-likelihood problems can be solved efficiently by means of the EM algorithm. Since (38) has the structure of a likelihood function, the EM algorithm may be applied at each SP iteration to compute the estimate  $\hat{\mathbf{b}}^{(m)}$  which maximizes  $\mu_{p \rightarrow \mathbf{b}}^{(m)}$ . Due to the similarity between objective functions (10) and (38), the application of the EM algorithm to the ML problem (36) leads to the same update parameter expressions as those defined in (16), (17), (18). However, due to the particular factorization of a priori probability  $p^{(m)}(\mathbf{a})$ , the computation of the required a posteriori probabilities  $p^{(m)}(a_k | \mathbf{r}, \hat{\mathbf{b}})$  is far more straightforward. Indeed, noticing that the matched-filter outputs are sufficient statistics for symbol detection, we may first write that

$$p^{(m)}(a_k | \mathbf{r}, \hat{\mathbf{b}}^{(m,n)}) = p^{(m)}(a_k | \mathbf{y}, \hat{\mathbf{b}}^{(m,n)}),$$

where  $\hat{\mathbf{b}}^{(m,n)}$  represents the  $n^{\text{th}}$  estimate generated by the EM algorithm at the  $m^{\text{th}}$  SP algorithm iteration and  $\mathbf{y}$  is the vector made up with matched filter outputs  $y_k(\nu, \tau)$ . Using Bayes rule, we get

$$p^{(m)}(a_k | \mathbf{r}, \hat{\mathbf{b}}^{(m,n)}) \sim \sum_{\sim \{a_k\}} p(\mathbf{y} | \mathbf{a}, \hat{\mathbf{b}}^{(m,n)}) p^{(m)}(\mathbf{a}).$$

Finally using the definition of  $p^{(m)}(\mathbf{a})$  and taking into account that the noise which affects the matched-filter outputs is white, we get

$$\begin{aligned} p^{(m)}(a_k | \mathbf{r}, \hat{\mathbf{b}}^{(m,n)}) &\sim \sum_{\sim \{a_k\}} \prod_l p(y_l | a_l, \hat{\mathbf{b}}^{(m,n)}) \mu_{a_l \rightarrow p}^{(m)}(a_l) \\ &\sim p(y_k | a_k, \hat{\mathbf{b}}^{(m,n)}) \mu_{a_k \rightarrow p}^{(m)}(a_k). \end{aligned} \quad (39)$$

A posteriori probabilities  $p^{(m)}(a_k | \mathbf{r}, \hat{\mathbf{b}}^{(m,n)})$  may therefore be simply computed by evaluating a Gaussian density and multiplying it by message  $\mu_{a_k \rightarrow p}^{(m)}$ .

Note that the derivation of the SP-EM synchronizer does not depend on the considered transmission scheme. The general SP-EM framework presented in this section may therefore be properly extended to the estimation of a wide range of parameters in various transmission schemes.

## VI. COMPARISON OF SP-EM AND EM SYNCHRONIZERS

In this section we compare the EM-based synchronization method presented in section III to the SP-EM synchronizer introduced in the previous section. In particular, we emphasize that the previously-proposed EM-based synchronization framework may be encompassed in the more general framework based on both the EM and the SP algorithms. The connection between the EM and the SP-EM synchronizers is shown in the case of a convolutionally-coded and a turbo-coded transmissions in the first two subsections.

Then, in the last subsection, we briefly address the computational complexity of these synchronization methods.

### *Convolutionally-coded Transmission*

The operations performed in the case of the EM and the SP-EM synchronizer are respectively summarized in Fig. 1 and 2. In the sequel, to avoid confusion, we will refer to the EM algorithm implemented at each iteration of the SP algorithm as  $\text{EM}_{\text{SP}}$  whereas the EM synchronizer presented in section III will simply be denoted by EM.

Let us first focus on the EM-based synchronizer. As mentioned above, the implementation of this iterative synchronizer requires the knowledge of posterior probabilities  $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)})$ . In the case of a convolutionally-coded transmission we may take benefit from the Markov structure underlying the transmitted signal to compute these posterior probabilities by means of the BCJR algorithm [21]. The EM synchronizer may then be implemented in an exact fashion.

Regarding the SP-EM approach, messages  $\mu_{a_k \rightarrow p}^{(m)}$  required for the implementation of the synchronizer may be computed by applying the SP algorithm to the factor graph associated with the code and the mapping used for the transmission. In the appendix, we derive the code-mapping factor-graph representation and we emphasize that applying the SP algorithm to this graph is equivalent to performing a decoding operation by means of a BCJR algorithm, i.e., messages passed by the SP algorithm along the graph edges are the same as those computed by a BCJR decoder. As a consequence, messages  $\mu_{a_k \rightarrow p}^{(m)}$  are actually equal to the so-called extrinsic probabilities  $p_e(a_k)$  delivered by a BCJR decoder. In particular, denoting by  $\hat{\mathbf{b}}_{\text{old}}$  the estimate computed at a given iteration, we have that posterior probabilities  $p(a_k|\mathbf{r}, \hat{\mathbf{b}}_{\text{old}})$  used by the EM synchronizer and extrinsic probabilities  $p_e(a_k)$  used by the SP-EM synchronizer at the next iteration are related according to

$$p(a_k|\mathbf{r}, \hat{\mathbf{b}}_{\text{old}}) \sim p(y_k|a_k, \hat{\mathbf{b}}_{\text{old}}) p_e(a_k). \quad (40)$$

Using then (39) and (40) we may notice that the SP-EM synchronizer defined in Fig. 2 reduces to the EM synchronizer defined in Fig. 1 if only one  $\text{EM}_{\text{SP}}$  iteration is performed at each SP iteration. The proposed SP-EM synchronizer may therefore be seen as a generalization of the previously-proposed EM-based synchronizer.

### Turbo-coded Transmission

The previous subsection addresses the implementation of the EM and the SP-EM synchronizers in a receiver which enables the exact computation of a posteriori probabilities  $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)})$ . In this subsection we consider the implementation of these iterative synchronizers in a turbo-coded system.

Let us first consider the EM synchronizer. As mentioned in Section III, a posteriori probabilities  $p(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)})$  required to implement the EM synchronizer are not available in turbo receivers. A common approach to deal with this problem is to approximate these probabilities by the so-called a posteriori probabilities  $\tilde{p}(a_k|\mathbf{r}, \hat{\mathbf{b}}^{(n-1)})$  delivered by the SISO modules constituting the turbo receiver. Making this approximation, the EM iterations are merged with the turbo ones and the implementation of the EM-based synchronizer does therefore not increase significantly the receiver complexity. Note however that the approximation on posterior probabilities is made without any guarantee of its validity. In particular, we are no longer ensured that the nice convergence properties of the EM algorithm still hold. In the sequel of this section we show for the case of a turbo-coded transmission that the EM-based approach may be viewed as an approximation of the proposed SP-EM synchronizer.

Let us now focus on the SP-EM synchronizer. Using turbo-code factor graph, messages  $\mu_{a_k \rightarrow p}^{(m)}$  required by the SP-EM synchronizer may be shown (see Appendix) to be related to extrinsic probabilities  $p_{e_1}^{(m)}(a_k)$  and  $p_{e_2}^{(m)}(a_k)$  delivered by the two SISO decoders at iteration  $m$  as

$$\mu_{a_k \rightarrow p}^{(m)} = \begin{cases} p_{e_1}^{(m)}(a_k) p_{e_2}^{(m)}(a_k) & \text{for systematic bits} \\ p_{e_i}^{(m)} & \text{for parity bits from encoder } i. \end{cases} \quad (41)$$

Notice to avoid confusion that, due to the particular message-passing schedule chosen in section IV, one turbo iteration corresponds to one SP algorithm iteration. Note also that extrinsic probabilities  $p_{e_1}^{(m)}(a_k)$  and  $p_{e_2}^{(m)}(a_k)$  are related [1] to the pseudo a posteriori probabilities  $\tilde{p}(a_k|\mathbf{r}, \hat{\mathbf{b}}_{\text{old}})$  used by the EM synchronizer according to

$$\tilde{p}(a_k|\mathbf{r}, \hat{\mathbf{b}}_{\text{old}}) \sim \begin{cases} p(y_k|a_k, \hat{\mathbf{b}}_{\text{old}}) p_{e_1}^{(m)}(a_k) p_{e_2}^{(m)}(a_k) & \text{for systematic bits} \\ p(y_k|a_k, \hat{\mathbf{b}}_{\text{old}}) p_{e_i}^{(m)}(a_k) & \text{for parity bits from encoder } i \end{cases} \quad (42)$$

where  $\hat{\mathbf{b}}_{\text{old}}$  denotes an estimate computed at the previous turbo iteration. Comparing then (42) to (39) by taking (41) into account, we may observe that SP-EM algorithm reduces to the classical EM approach if only one  $\text{EM}_{\text{SP}}$  iteration is performed at each SP iteration. Let us however insist on the fact that the SP-EM synchronizer is not a trivial extension of the EM synchronizer i.e. the iterative nature of the SP-EM synchronizer (at a given turbo iteration) cannot be justified by standard EM arguments.

Note also that the method proposed by Zhang et al. [5]-[7] may be understood in the factor-graph framework by the previous reasoning. Indeed, the authors of [5]-[7] use extrinsic probabilities as a priori probabilities to build a (modified) likelihood function. The objective function that Zhang et al. propose to maximize at each iteration is therefore very similar to the one derived from the SP algorithm; the only difference being that, instead of using (41), Zhang et al. only use the extrinsic probability from the second decoder as a priori probability of the corresponding systematic bit. The maximization method proposed in [5]-[7] implies approximations on the likelihood function (33) whereas in this paper we propose to perform the maximization of (33) by means of the  $EM_{SP}$  algorithm.

#### *Comparison of synchronizer complexity*

In this subsection we briefly discuss the computational complexity of the receiver when it is synchronized by either the EM or the SP-EM approach. Tab. I contains a rough evaluation of the number of additions and multiplications required to performed the basic receiver operations. These numbers are discussed and explained in the remainder of the subsection.

*a) Demapping/Decoding:* The computational complexity of the demapper and the decoder may be evaluated by considering message update equations (47), (48), (49) and (51) in the Appendix. In Tab. I,  $Q$  denotes the memory of the code and  $K_c$  is a factor whose value depends on the kind of code which is considered for the transmission:  $K_c = 1$  for a convolutional code,  $K_c = 2$  for a turbo code. Results in Tab. I represent the number of additions and multiplications required for one demapping/decoding operation. If the considered receivers are iterative, these expressions have to be multiplied by the number of iterations, say  $N_{it}^{dec}$ , which has to be performed to achieved the required performance.

*b) Synchronization:* Since it has been shown that the EM synchronizer is a particular case of the SP-EM approach, we consider in Tab. I the number of operations required to perform one  $EM_{SP}$  iteration. If we perform several  $EM_{SP}$  iterations at each SP iteration, say  $N_{it}^{EM}$ , results in Tab. I have simply to be multiplied by  $N_{it}^{EM}$ .

Notations  $N_{MF}^a$  and  $N_{MF}^m$  represent the number of additions and multiplications which have to be performed to compute one matched-filter output  $y_k(\nu, \tau)$  at each  $EM_{SP}$  iteration. These parameters vary as a function of the considered synchronization parameter. In the case of the channel gain or the carrier phase offset estimation, the matched-filter outputs do not have to be recomputed at each  $EM_{SP}$  iteration

and therefore  $N_{MF}^a = 0$  and  $N_{MF}^m = 0$ . For timing estimation,  $y_k(\nu, \tau)$  may be computed by using an interpolator operating on a sampled version of the matched filter output [4]. In this case,  $N_{MF}^a$  and  $N_{MF}^m$  depend on the considered interpolator. In general,  $N_{MF}^a \simeq 4$  and  $N_{MF}^m \simeq 4$  enables a sufficient accuracy for the considered SNRs. Finally, in the case of carrier frequency estimation, matched-filter outputs  $y_k(\nu, \tau)$  should in theory be recomputed at each iteration from received samples  $r(kT_s)$ . In that case,  $N_{MF}^a$  and  $N_{MF}^m$  may be quite large (around 20). Note however that this operation may be avoided if  $\nu \ll \frac{1}{T}$ , in which case  $N_{MF}^a = 0$  and  $N_{MF}^m = 0$  (See [4] for more details).

From (16)-(18), we may also notice that the timing or the frequency offset estimation require to solve a maximization problem. Notations  $N_{max}^a$  and  $N_{max}^m$  in Tab. I represent the number of additions and multiplications which have to be performed to solve these maximization problems.

*c) Overall Receiver Complexity:* In the evaluation of the receiver overall complexity, we only consider the number of required multiplications since this operation is much more complex to perform than an addition. Combining the operations required for demapping/decoding and synchronizing in Tab. I, we get

$$\begin{aligned}
 & \underbrace{N_{it}^{dec} \left[ K|\mathcal{A}| (\log_2^2 |\mathcal{A}| - 1) + K_c K \log_2 |\mathcal{A}| (2^{Q+3} R + 2) \right]}_{\text{Demapping/Decoding}} \\
 & + \underbrace{N_{it}^{dec} N_{it}^{EM} \left[ K(N_{MF}^m + 2) + N_{max}^m \right]}_{\text{Synchronization}}. \tag{43}
 \end{aligned}$$

In (43), the underbraces indicate how each receiver task affects the overall receiver complexity. It is clear from (43) that increasing the number of EM<sub>SP</sub> iterations, i.e.  $N_{it}^{EM}$ , will increase the complexity of the synchronization part. However, this increase of the synchronization complexity may be compensated by a reduction of the number of SP iterations, i.e.  $N_{it}^{dec}$ , required to achieve some given performance. Actually, the optimal number of EM<sub>SP</sub> iterations which should be performed at each SP iteration to minimize the overall receiver complexity depends on the considered code, synchronization parameters to estimate, etc. The optimal tuning of parameter  $N_{it}^{EM}$  is however beyond the scope of this paper and will therefore not be considered hereafter.

## VII. SIMULATION RESULTS

In this section we illustrate the performance of the EM and the SP-EM synchronizers by means of simulation results. We consider the synchronization of the carrier phase offset and the timing epoch in

the case of a convolutionally-coded and a turbo-coded transmission. The carrier frequency offset  $\nu$  and the channel gain are assumed to be known. The system performance is assessed in terms of mean square error (MSE) and bit-error rate (BER).

Note that, depending on the considered message-update schedule, the SP-EM approach may lead to two slightly-different algorithms. Indeed, at the first iteration we may either update messages  $\mu_{a_k \rightarrow p}$  first by taking into account an initial estimate  $\hat{\mathbf{b}}^{(0)}$  or compute an estimate  $\hat{\mathbf{b}}^{(1)}$  first by taking into account initial messages  $\mu_{a_k \rightarrow p}^{(0)}$ . Both approaches will be considered in sequel: the approach in which we first update the parameter estimate will be referred to as SP-EM1 whereas the approach in which we first perform a decoding operation will be referred to as SP-EM2. Note also that the SP-EM approaches are parameterized by the number of  $\text{EM}_{\text{SP}}$  iterations performed at each SP iteration. In the sequel we will use the notation  $\text{SP-EM}\{N\}$  to indicate that  $N$   $\text{EM}_{\text{SP}}$  iterations are performed at each SP iteration.

### *Convolutionally-coded Transmission*

We consider a rate-1/3 non-systematic convolutional (NSC) encoder with generator polynomials  $(25, 33, 37)_8$ . At the receiver we consider a BCJR decoder. The  $E_b/N_0$ -ratio is set to 2.5dB. Transmitted frames consist of 252 QPSK symbols. The roll-off factor  $\alpha$  is equal to 0.2. The carrier phase offset  $\theta$  and the timing offset  $\tau$  are assumed to be uniformly distributed respectively on the interval  $[-0.5T, 0.5T]$  and  $[-\pi/4, \pi/4]$ , and they are changed for each new transmitted frame. All the iterative synchronizers are initialized with a phase and a timing estimate respectively computed via a Viterbi&Viterbi (VV) [18] and an Oerder&Meyr (OM) [19] synchronizer. We assume that both the frame synchronization and the phase ambiguity problems are perfectly solved i.e. at each iteration the timing estimate  $\hat{\tau}$  is constrained to be such that  $|\tau - \hat{\tau}| \leq 0.5T$  and the phase estimate is such that  $|\theta - \hat{\theta}| < \pi/4$ .

Fig. 3-5 illustrate the speed of convergence in terms of MSE and BER when the system is synchronized according to different methods. The BER performance is represented versus the number of multiplications performed by the system (see (43)). The MSE is represented versus the number of decoding operations. The iteration labeled “0” in the MSE plots represents the estimation quality which is achieved before entering the decoder for the first time. Let us consider the MSE achieved by the synchronizers at this iteration. On the one hand we see that the EM and the SP-EM2 synchronizers exhibit the same performance as the conventional NDA synchronizers. Indeed, these synchronizers do not update the parameter estimate before entering the decoder and therefore simply deliver the same estimate as the

Viterbi&Viterbi and the Oerder&Meyr synchronizers. On the other hand, we observe that the SP-EM1 is able to improve the MSE before performing any decoding operation. In fact, at iteration 0 the SP-EM1 tries to solve a ML NDA problem by means of the  $EM_{SP}$  algorithm. Indeed, a common approach to initialize messages  $\mu_{a_k \rightarrow p}^{(0)}$  is to choose

$$\mu_{a_k \rightarrow p}^{(0)}(a) = 1/M \quad \forall a \in \mathcal{A}, \quad (44)$$

i.e. to assume that all the transmitted symbols (and consequently all the possible transmitted sequences) are equiprobable. Comparing this assumption with (11), we see that the function (33) maximized by the SP-EM1 algorithm at iteration 0 is nothing but the NDA likelihood function. The difference between the SP-EM1 performance and the conventional NDA synchronizer performance represents therefore the performance degradation due to the approximation on which are based the Viterbi&Viterbi and the Oerder&Meyr synchronizers. We see that the Viterbi&Viterbi operates very close to the ML NDA performance whereas the Oerder&Meyr leads to more degradations.

We may also notice that, at the considered low SNR, the conventional NDA synchronizers operate quite far from the modified Cramer-Rao bound (MCRB) [24] and do not enable to recover the same BER as the perfectly-synchronized system. The considered iterative synchronizers enable to reach both the MCRB and the BER of the perfectly-synchronized system after a sufficient number of iterations. We may notice that the SP-EM approaches enable to slightly decrease the receiver complexity: the BER of the perfectly-synchronized is achieved with less multiplications<sup>2</sup> when using the SP-EM synchronizers than the EM synchronizer.

### *Turbo-coded Transmission*

In this section we deal with the joint phase and timing synchronization of a turbo-coded transmission. We consider a turbo encoder which consists of two rate-1/2 RSC encoders with polynomial generators  $(37, 33)_8$ , separated by an interleaver. The turbo encoder output is punctured so that the overall code rate is 1/2. The roll-off factor  $\alpha$  is set to 0.1. This choice for  $\alpha$  is quite aggressive for timing estimation but will allow us to clearly emphasize the difference between the compared synchronizers. Note however that larger values of the roll-off factor attenuate the difference between the different synchronizer performance. Transmitted frames consist of 512 BPSK symbols. The timing and the phase offset are changed for

<sup>2</sup>In our simulations, maximization problem (16) is solved via a Newton-Raphson method and required derivatives are evaluated numerically. Therefore,  $N_{MF}^m = 5$  and  $N_{max}^m \simeq 3K(N_{MF}^m + 2)$ ,  $Q = 4$  and  $\log_2 |\mathcal{A}| = 2$ .

each new transmitted frame and are assumed to be uniformly distributed respectively on  $[-0.5T, 0.5T]$  and on  $[-\pi/2, \pi/2]$ . We assume that the frame synchronization and phase ambiguity problems are perfectly solved, i.e., at each iteration the timing and the phase estimates are constrained to be such that  $|\tau - \hat{\tau}| \leq 0.5T$  and  $|\theta - \hat{\theta}| < \pi/2$ .

Fig. 6 and 7 respectively show the MSE for timing estimation and the BER achieved with different synchronizers versus the  $E_b/N_0$ -ratio. In addition to the iterative EM and SP-EM synchronizers, we consider the conventional Viterbi&Viterti and Oerder&Meyr synchronizers. We perform 10 turbo iterations and the iterative synchronizers are initialized with initial estimates computed by the OM and VV estimators. The behavior of the MSE associated with phase estimation is similar to the one of timing estimation and is not shown here.

As in the case of a convolutionally-coded transmission, we may observe the large gap between the MSE achieved by the iterative synchronizers and the MSE achieved by the conventional OM estimator. As mentioned above, this difference may be explained by the NDA nature of the OM synchronizer and the fact that it relies on a low-SNR approximation of the actual likelihood function.

Note also from Fig. 6 that SP-EM1 and SP-EM2 outperform the classical EM synchronizer whatever the  $E_b/N_0$ -ratio and reach the MCRB for sufficiently-high SNRs. The conclusions drawn for the MSE may be translated in terms of BER. We see in Fig. 7 that the OM which exhibits the poorest estimation quality also leads to the largest BER degradation. The SP-EM-based approaches which reach a very good estimation quality also enable to achieve a BER very close to the performance of the perfectly synchronized system. The EM synchronizer improves the BER with respect to the OM synchronizer but does not recover the performance of the perfectly synchronized system after 10 iterations. We will see in the sequel that this gap is due to the slow convergence of the EM synchronizer. In Fig. 8-10 we illustrate the speed of convergence achieved by the system with different synchronizers. The MSE is represented in Fig. 8-9 versus the number of turbo iterations. Again, iteration "0" represents the MSE achieved by the synchronizer before performing any decoding iteration. The BER is illustrated in Fig. 10 versus the number of multiplications required by the system. The considered  $E_b/N_0$ -ratio is equal to 2.5dB.

Let us first focus on the performance in terms of MSE. Concerning phase estimation, we may notice that the performance achieved by the VV synchronizer is very close to the MCRB. Unlike the VV synchronizer, the OM synchronizer performance is quite far from the MCRB and therefore prevent the

convergence of the turbo decoder as shown from the BER figure. Again, it is clear that the iterative receivers enable to improve the estimation quality and the MCRB is reached for a sufficient number of iterations. We may note the importance of the initial estimate: the better the initial estimate, the faster the convergence (in terms of turbo iterations). The SP-EM1 approach does therefore always outperform the SP-EM2 approach. Notice also that the system convergence is all the faster as the number of  $EM_{SP}$  iterations performed per SP iteration is larger. In particular, the EM synchronizer which is actually equivalent to the case SP-EM2{1} exhibits a very slow convergence.

Note, however, that increasing the number of  $EM_{SP}$  iterations also increases the complexity. This may be visualized by means of the BER plot in Fig. 10. We see that increasing the number of  $EM_{SP}$  iterations performed per SP iteration decreases the overall system complexity up to a point: *i)* the EM synchronizer (which is equivalent to the case SP-EM2{1}) has the poorest performance and requires a huge number of multiplications to achieve the BER of a perfectly-synchronized system; *ii)* the SP-EM synchronizers enable to reduce the number of required multiplications. However, performing 5  $EM_{SP}$  iterations per SP iteration seems to lead to better performance than performing 15  $EM_{SP}$  iterations. In fact, the additional complexity introduced by the SP-EM{15} is not compensated by a sufficient decrease of the BER. In other words, the SP-EM{5} makes the best compromise between the number of  $EM_{SP}$  and SP iterations to perform.

## VIII. CONCLUSION

In this paper, we place the synchronization of a digital receiver into the factor graph and the sum-product (SP) algorithm framework. General equations are derived and some previously-proposed iterative synchronization methods [5]-[9] are shown to have an interpretation in the factor-graph framework. In particular, we propose an iterative synchronization framework based on both the SP and the EM algorithms. We then emphasize that a previously-proposed iterative synchronizer based only on the EM algorithm [9] may actually be viewed as a particular case of the SP-EM synchronization method proposed in this paper. The performance of the proposed synchronizer is finally assessed in terms of mean-square error and bit-error rate for convolutionally-coded and turbo-coded transmissions. It is shown in the case of joint phase and timing estimation that the proposed approach outperforms classical synchronization methods.

## ACKNOWLEDGMENT

This work has been supported by the Interuniversity Attraction Poles Program P5/11 - Belgian State - Federal Office for Scientific, Technical and Cultural Affairs.

## APPENDIX

In this appendix we develop the factor graph of  $p(\mathbf{a})$  in different scenarios. Transmitted symbols  $\mathbf{a}$  result from a coding and a mapping operation. Taking this fact into account we may rewrite  $p(\mathbf{a})$  as

$$\begin{aligned} p(\mathbf{a}) &= \sum_{\sim \mathbf{a}} p(\mathbf{a}, \mathbf{x}, \mathbf{u}), \\ &= \sum_{\sim \mathbf{a}} p(\mathbf{a}|\mathbf{x}) p(\mathbf{x}, \mathbf{u}). \end{aligned} \quad (45)$$

Factor  $p(\mathbf{a}|\mathbf{x})$  relates the coded bits to the transmitted symbols and accounts therefore for the mapping operation. Factor  $p(\mathbf{x}, \mathbf{u})$  accounts for the relation between the input and the output of the decoder. We consider separately the factor-graph representation of these two factors.

*Mapper*

Denoting by  $\tilde{\mathbf{x}}_k$  the group of coded bits mapped onto symbol  $a_k$ , we have that probability  $p(\mathbf{a}|\mathbf{x})$  simply writes

$$p(\mathbf{a}|\mathbf{x}) = \prod_k \mathbb{I}\{a_k = \text{out}(\tilde{\mathbf{x}}_k)\}, \quad (46)$$

where  $\mathbb{I}\{\cdot\}$  denotes the indicator function, which is equal to 1 if the statement between the braces is true and equal to 0 otherwise, and  $\text{out}(\tilde{\mathbf{x}}_k)$  corresponds to the mapper output if  $\tilde{\mathbf{x}}_k$  is input. The factor graph representation of one factor of (46) is shown in Fig. 12. Denoting by  $\mu_{\tilde{\mathbf{x}}_k, i \rightarrow f_k}$  (resp.  $\mu_{f_k \rightarrow \tilde{\mathbf{x}}_k, i}$ ) the messages outgoing (resp. entering) coded-bit nodes, we have

$$\mu_{f_k \rightarrow a_k}(a_k) = \sum_{\sim a_k} \mathbb{I}\{a_k = \text{out}(\tilde{\mathbf{x}}_k)\} \prod_i \mu_{\tilde{\mathbf{x}}_k, i \rightarrow f_k}(\tilde{x}_{k,i}), \quad (47)$$

$$\begin{aligned} \mu_{f_k \rightarrow \tilde{\mathbf{x}}_k, j}(\tilde{x}_{k,j}) &= \sum_{\sim \tilde{\mathbf{x}}_k, j} \mathbb{I}\{a_k = \text{out}(\tilde{\mathbf{x}}_k)\} \mu_{a_k \rightarrow f_k}(a_k) \\ &\quad \times \prod_{i \neq j} \mu_{\tilde{\mathbf{x}}_k, i \rightarrow f_k}(\tilde{x}_{k,i}), \end{aligned} \quad (48)$$

where  $\tilde{x}_{k,j}$  is the  $j^{\text{th}}$  bit of transmitted symbol  $a_k$ .

### Convolutional Code

We now consider the factorization of probability  $p(\mathbf{x}, \mathbf{u})$  in the case of a convolutional code. Let us first notice that the output of a convolutional code may be regarded as the output of a Markov process. Defining then vector  $\mathbf{s} = [s_0, s_1, \dots, s_L]^T$  as the vector of state variables in the Markov model, we may write  $p(\mathbf{x}, \mathbf{u})$  as follows:

$$\begin{aligned} p(\mathbf{x}, \mathbf{u}) &= \sum_{\sim \mathbf{s}} p(\mathbf{s}, \mathbf{x}, \mathbf{u}), \\ &= \sum_{\sim \mathbf{s}} \mathbb{I}\{s_0 = 0\} \prod_l \mathbb{I}\{s_{l+1} \leftarrow (u_l, s_l)\} \\ &\quad \times \mathbb{I}\{\mathbf{x}_l = \text{out}(u_l, s_l)\} \end{aligned} \quad (49)$$

where function  $\text{out}(u_l, s_l)$  corresponds to the coder output for input  $u_l$  and current state  $s_l$ . The factor graph representation of one factor of (49) is shown in Fig. 13. Using the message notations from Fig. 13, we have that the message entering the coded-bit node is equal to

$$\begin{aligned} \mu_{f_c \rightarrow x_{l,j}}(x_{l,j}) &= \sum_{\sim x_{l,j}} \alpha_l(s_l) \beta_{l+1}(s_{l+1}) \prod_{i \neq j} \mu_{x_{l,i} \rightarrow f_c}(x_{l,i}) \\ &\quad \times \mathbb{I}\{s_{l+1} \leftarrow (u_l, s_l)\} \mathbb{I}\{\mathbf{x}_l = \text{out}(u_l, s_l)\}, \end{aligned} \quad (50)$$

where  $x_{l,i}$  denotes the  $i^{\text{th}}$  coded bit delivered at time  $l$ . In [3], it is shown that messages  $\alpha_l$  and  $\beta_l$  computed by the SP algorithm are equal to the well-known  $\alpha$  and  $\beta$  recursions performed by the BCJR algorithm. Therefore, comparing (50) with the definition of the so-called extrinsic probabilities in [1], we may notice that message  $\mu_{f_c \rightarrow x_{l,j}}$  is actually equal to the extrinsic probabilities of coded bit  $x_{l,j}$ . Following the same reasoning it is straightforward to show that message  $\mu_{f_l \rightarrow u_l}$  is equal to the extrinsic probabilities of information bit  $u_l$ .

### Turbo Code

We consider the factor graph of an unpunctured turbo code made up with two rate-1/2 recursive convolutional encoders separated by an interleaver. Let  $u_{\pi(l)}$  denote the  $l^{\text{th}}$  information bit of the interleaved sequence. Since the turbo code is made up with two convolutional encoders, probability

$p(\mathbf{x}, \mathbf{u})$  may be expressed as

$$\begin{aligned}
p(\mathbf{x}, \mathbf{u}) = & \sum_{\sim \mathbf{s}} \mathbb{I}\{s_0^1 = 0\} \prod_l \mathbb{I}\{s_{l+1}^1 \leftarrow (u_l, s_l^1)\} \\
& \times \mathbb{I}\{\mathbf{x}_l^1 = \text{out}(u_l, s_l^1)\} \\
& \times \mathbb{I}\{s_0^2 = 0\} \prod_l \mathbb{I}\{s_{l+1}^2 \leftarrow (u_{\pi(l)}, s_l^2)\} \\
& \times \mathbb{I}\{\mathbf{x}_l^2 = \text{out}(u_{\pi(l)}, s_l^2)\}, \tag{51}
\end{aligned}$$

where  $s_l^i$  and  $\mathbf{x}_l^i$  respectively represent the state and the coded bits relative to code  $i$  at time  $l$ . Factor graph representation of (51) is shown in Fig. 14. In this figure, the factor graphs relative to the two constituent convolutional code are clearly visible. Using the result from the previous subsection, messages  $\mu_{f_i^1 \rightarrow u_i}$  and  $\mu_{f_i^1 \rightarrow x_i^1}$  computed by applying the SP algorithm on the factor graph relative to convolutional code 1 are respectively equal to the extrinsic probabilities of information bit  $u_i$  and coded bit  $x_i^1$ . In the same way, messages  $\mu_{f_i^2 \rightarrow u_{\pi(i)}}$  and  $\mu_{f_i^2 \rightarrow x_i^2}$  are respectively equal to the extrinsic probabilities of information bit  $u_{\pi(i)}$  and coded bit  $x_i^2$ . Applying then the SP-algorithm message update rules, we have that the messages leaving the code factor graph are simply equal to

$$\mu_{out}(u_i) = \mu_{f_i^1 \rightarrow u_i}(u_i) \mu_{f_{\pi(i)}^2 \rightarrow u_i}(u_i), \tag{52}$$

$$\mu_{out}(x_i^1) = \mu_{f_i^1 \rightarrow x_i^1}(x_i^1), \tag{53}$$

$$\mu_{out}(x_i^2) = \mu_{f_i^2 \rightarrow x_i^2}(x_i^2), \tag{54}$$

where  $f_{\pi(i)}^2$  is the function which corresponds to information bit  $u_i$  in the factor graph of decoder 2.

## REFERENCES

- [1] C. Berrou, A. Glavieux and P. Thitimajshima. "Near Shannon limit error-correcting coding and decoding". In *IEEE International Conference on Communications, ICC'*, pages pp. 1064–1070, Geneva, Switzerland, May 1993.
- [2] G. Caire, G. Taricco and E. Biglieri. "Bit-interleaved coded modulation". *IEEE Trans. on Inform. Theory*, 44:pp. 927–946, May 1998.
- [3] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. on Inform. Theory*, 47:pp. 498–519, February 2001.
- [4] H. Meyr, M. Moeneclaey, and S. Fatchel. *Digital Communication Receivers : Synchronization, Channel Estimation and Signal Processing*. Wiley Series in Telecommunications and Signal Processing, USA, 1998.
- [5] L. Zhang and A. Burr. "A Novel Carrier Phase Recovery Method for Turbo Coded QPSK systems". In *European Wireless, EW'*, Florence, Italy, Feb. 2000.
- [6] L. Zhang and A. Burr. "APPA Symbol Timing Recovery Scheme for Turbo-codes". In *IEEE International Symposium on Personal Indoor and Mobile Radio Communications, PIMRC'*, Lisbonne, Portugal, Nov. 2002.

- [7] L. Zhang and A. Burr. "Application of Turbo Principle to Carrier Phase Recovery in Turbo Encoded Bit-Interleaved Coded Modulation System". In *Int. Symp. on Turbo Codes & Rel. Topics, ISTC'*, Brest, France, Sept. 2003.
- [8] V. Lottici and M. Luise. Carrier phase recovery for turbo-coded linear modulations. In *IEEE International Conference on Communications, ICC'*, New-York, USA, April 2002.
- [9] N. Noels, C. Herzet, A. Dejonghe, V. Lottici, H. Steendam, M. Moeneclaey, M Luise and L. Vandendorpe. "Turbo-synchronization: an EM algorithm approach". In *Proc. IEEE ICC*, Anchorage, May 2003.
- [10] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *J. Roy. Stat. Soc.*, 39(1):pp. 1–38, January 1977.
- [11] C. Herzet, V. Ramon, L. Vandendorpe and M. Moeneclaey. "EM algorithm-based timing synchronization in turbo receivers". In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'*, Hong Kong, April 2003.
- [12] V. Ramon, C. Herzet, L. Vandendorpe and M. Moeneclaey. "EM algorithm-based multiuser synchronization in turbo receivers". In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'*, Montreal, April 2004.
- [13] L. Sutskever, S. Shamai and J. Ziv. "A novel approach to iterative joint decoding and phase estimation". In *Int. Symp. on Turbo Codes & Rel. Topics, ISTC'*, Brest, France, September 2003.
- [14] J. Dauwels and H.-A. Loeliger. "Phase Estimation by Message Passing". In *IEEE International Conference on Communications, ICC'*, Paris, France, June 2004.
- [15] J.R. Barry, A. Kavcic, S.W. McLaughlin, A. Nayak and W. Zeng. "Iterative Timing Recovery". *IEEE Signal Processing Mag.*, pages pp. 89–102, January 2004.
- [16] J.G. Proakis. *Digital Communications-3rd ed.* McGraw-Hill International Editions, 1995.
- [17] J.M. Mendel. *Lessons in Estimation Theory for Signal Processing Communications and Control.* Prentice Hall Signal Processing Series, Englewood Cliffs, NJ, 1995.
- [18] A. J. Viterbi and A. M. Viterbi. "Nonlinear estimation of PSK-Modulated Carrier Phase with Application to Burst Digital Transmission". *IEEE Trans. on Inform. Theory*, 29:pp. 543–551, Jul. 1983.
- [19] M. Oerder and H. Meyr. "Digital filter and square timing recovery". *IEEE Trans. Comm.*, 36:605–611, May 1988.
- [20] C. F. J. Wu. On the convergence properties of the EM algorithm. *Ann. Statistics*, 11(1):pp. 95–103, 1983.
- [21] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate". *IEEE Trans. on Inform. Theory*, 20:pp. 284–287, March 1974.
- [22] T.M. Cassaro and C.N. Georghiades. "Frame synchronization for coded systems over AWGN channel". 52(2):pp. 484–489, March 2004.
- [23] H. Wymeersch and M. Moeneclaey. Iterative Code-Aided ML Phase Estimation and Phase Ambiguity Resolution. 2005(6), 2005.
- [24] A. D'Andrea, U. Mengali and R. Reggiannini. "The Modified Cramer-Rao Bound and Its Application to Synchronization Problems". *IEEE Trans. Comm.*, 42(2/3/4):pp. 1391–1399, February/March/April 1994.

```
1  $\hat{\mathbf{b}}^0 = \mathbf{b}_0$ ;  
   for  $n = 1 \rightarrow N$  do  
2   Computation of  $p(a_k | \mathbf{r}, \hat{\mathbf{b}}^{(n-1)}) \quad \forall a_k$ ;  
3   Computation of  $\hat{\mathbf{b}}^{(n)}$  (see (15)-(18));  
   end
```

Fig. 1: Summary of the operations performed by the EM synchronizer.

```

1  $\hat{\mathbf{b}}^{0,N} = \mathbf{b}_0;$ 
  for  $m = 1 \rightarrow M$  (SP iterations) do
2    $\hat{\mathbf{b}}^{m,0} = \hat{\mathbf{b}}^{(m-1,N)};$ 
3   Computation of  $\mu_{a_l \rightarrow p_l}^{(m)}(a_l)$  (see Appendix);
   for  $n = 1 \rightarrow N$  (EMSP iterations) do
4     Computation of  $p(a_k | \mathbf{r}, \hat{\mathbf{b}}^{(m,n-1)}) \forall a_k$  (see (39));
5     Computation of  $\hat{\mathbf{b}}^{(m,n)}$  (see (15)-(18));
   end
end

```

Fig. 2: Summary of the operations performed by the SP-EM synchronizer.

	<i>Additions</i>	<i>Multiplications</i>
<i>Mapper</i>	$K \log_2  \mathcal{A}  ( \mathcal{A}  - 2)$	$K \mathcal{A}  (\log_2^2  \mathcal{A}  - 1)$
<i>Decoder</i>	$K_c K \log_2  \mathcal{A}  2^{Q+1} (2R + 1)$	$K_c K \log_2  \mathcal{A}  (2^{Q+3} R + 2)$
<i>Synchro</i>	$\simeq K(N_{MF}^a +  \mathcal{A} ) + N_{max}^a$	$K(N_{MF}^m + 2) + N_{max}^m$

TABLE I

NUMBER OF ADDITIONS AND MULTIPLICATIONS REQUIRED TO PERFORM DIFFERENT OPERATIONS AT THE RECEIVER.

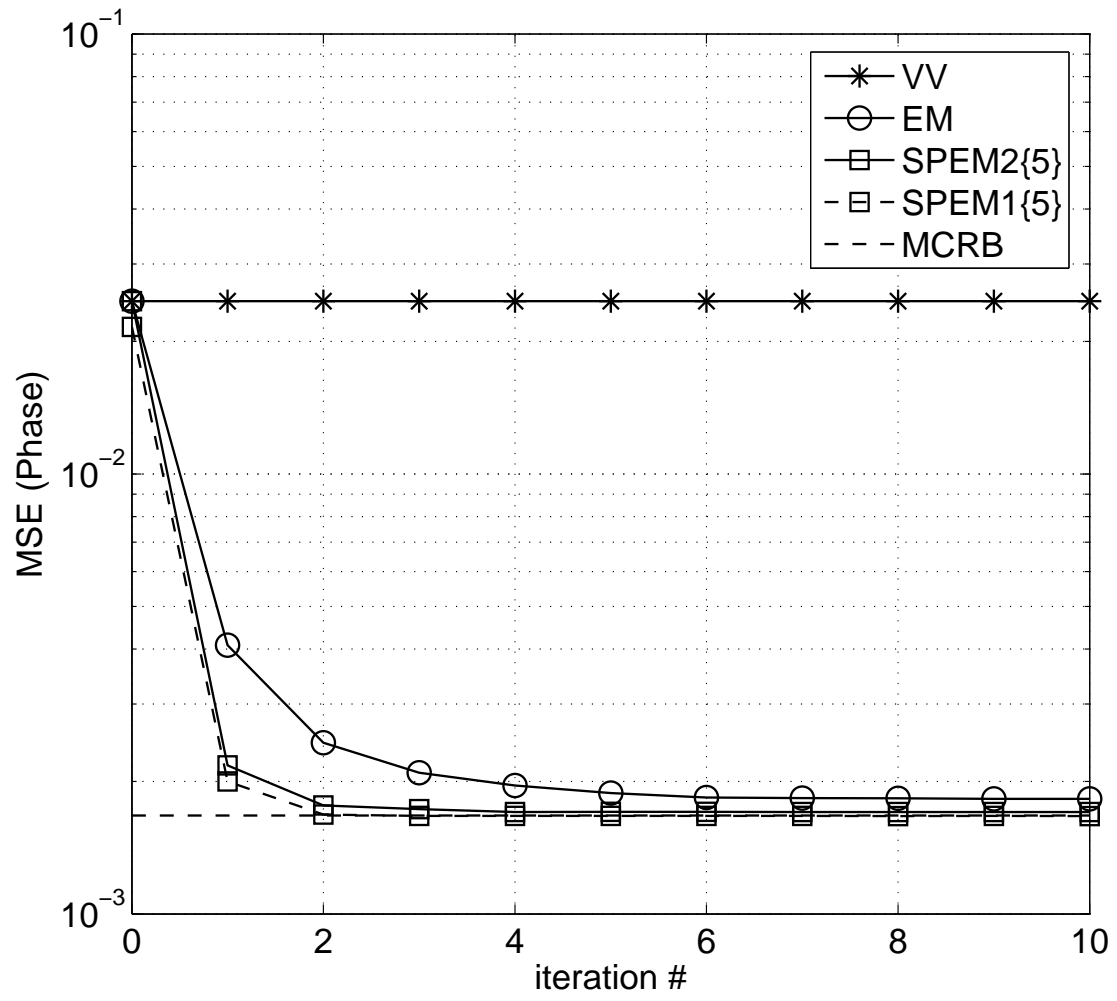


Fig. 3. MSE for phase estimation and convolutionally-coded transmission.

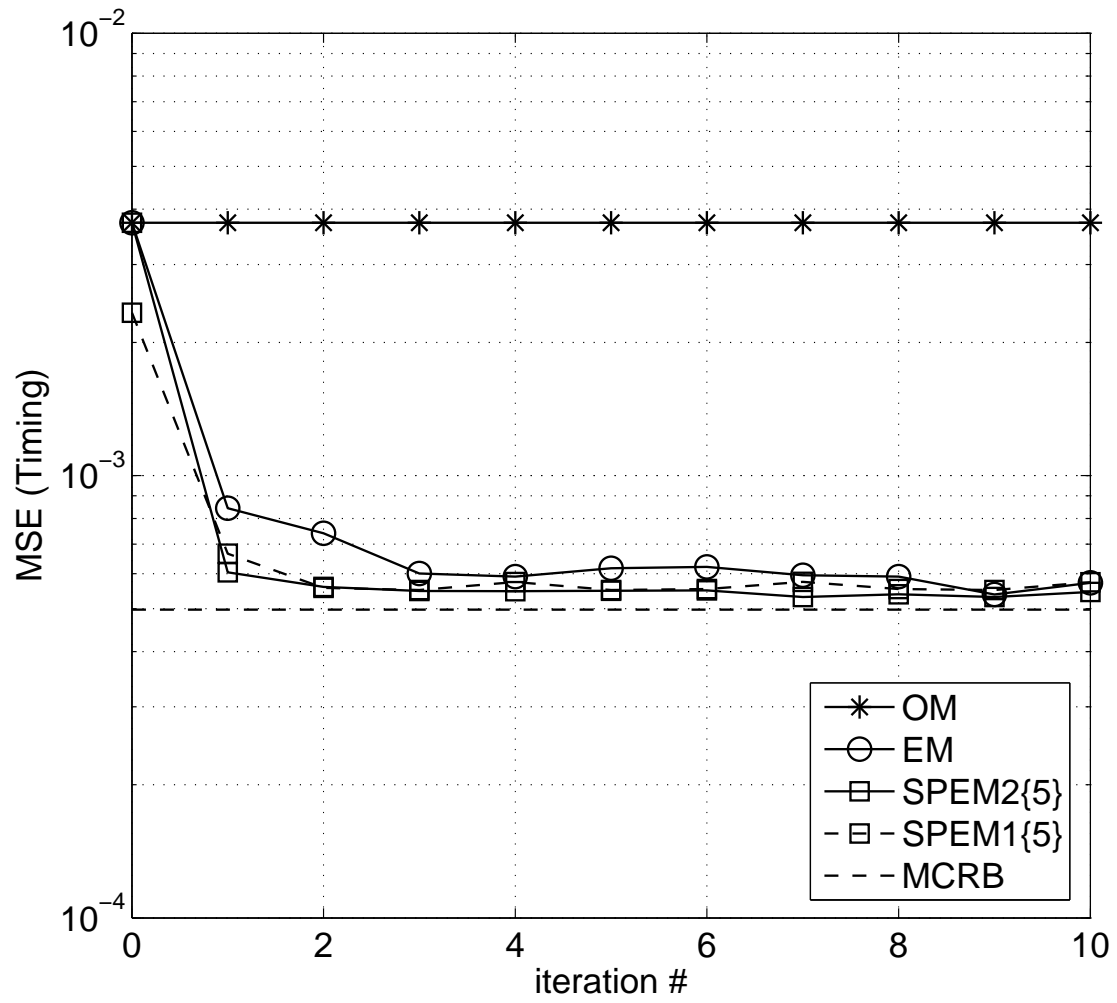


Fig. 4. MSE for timing estimation and convolutionally-coded transmission.

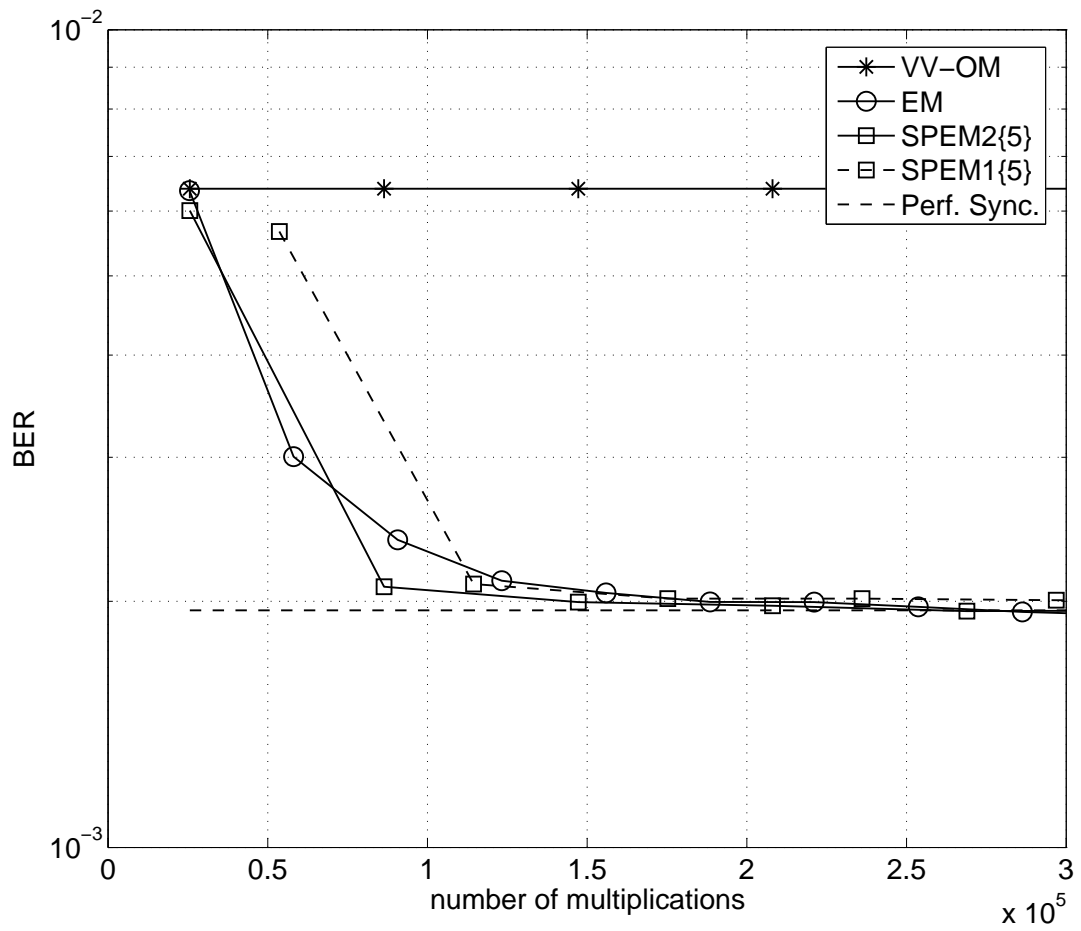


Fig. 5. BER achieved by the convolutionally-coded system versus the number of required multiplications.

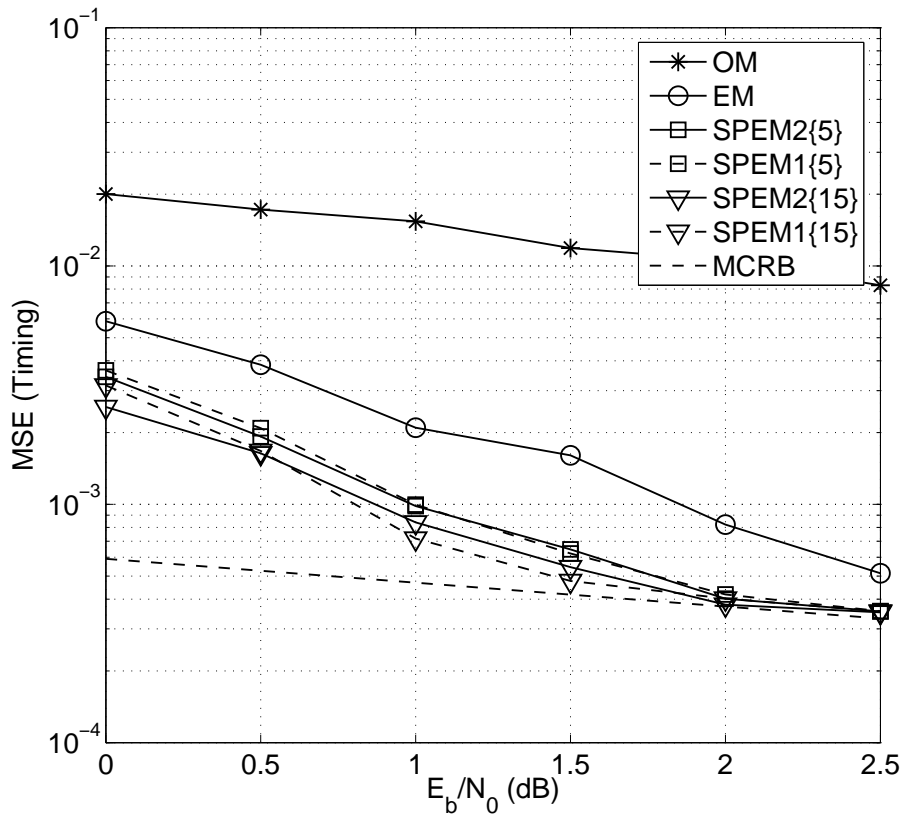


Fig. 6. MSE for timing estimation versus  $E_b/N_0$ -ratio for a turbo-coded transmission.

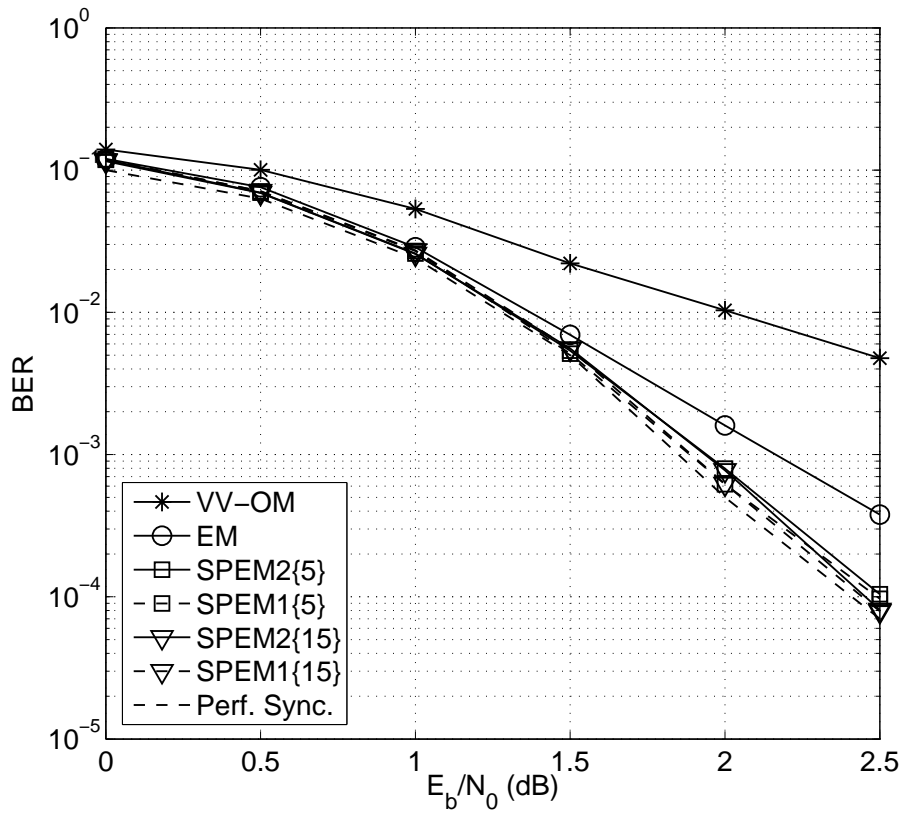


Fig. 7. BER versus  $E_b/N_0$ -ratio for a turbo-coded transmission.

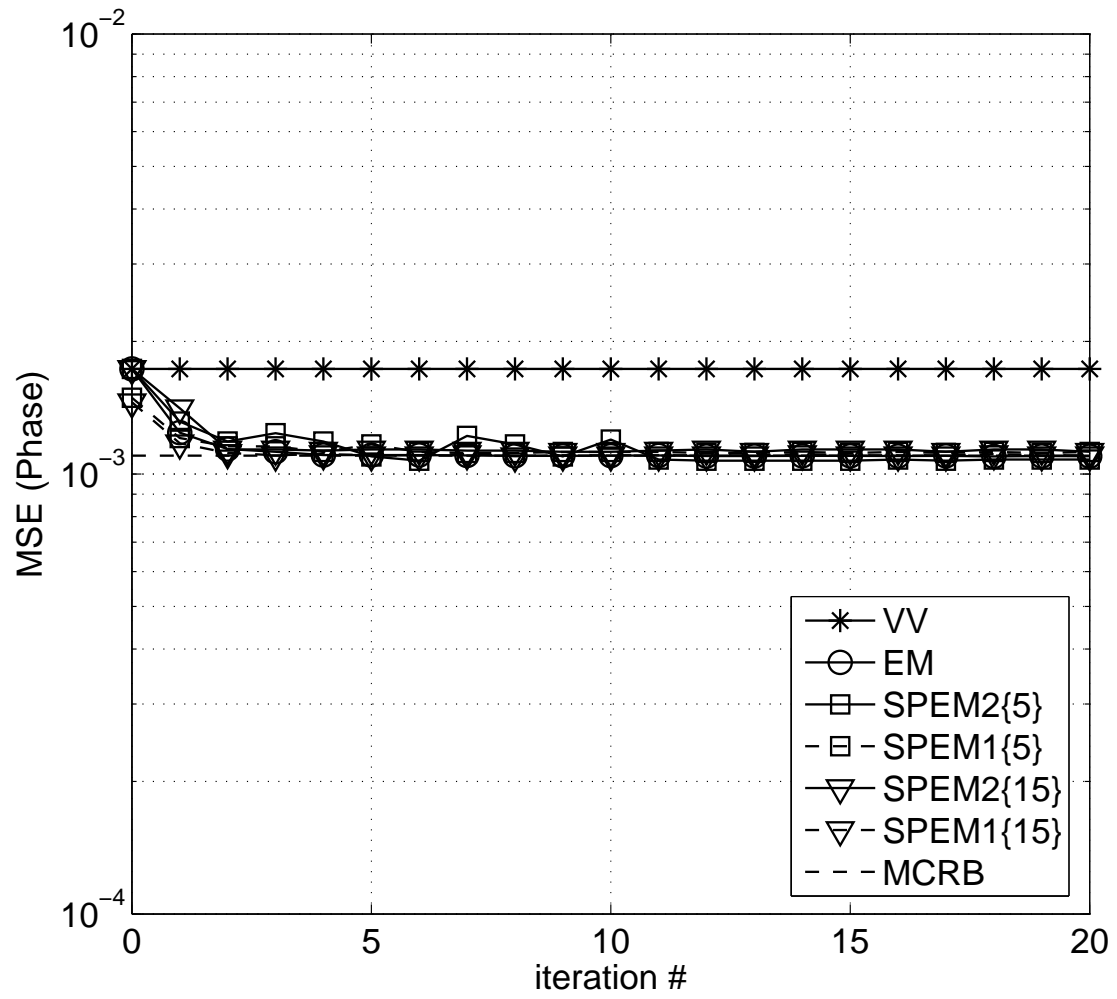


Fig. 8. MSE for phase estimation and turbo-coded transmission versus the number of turbo iterations.

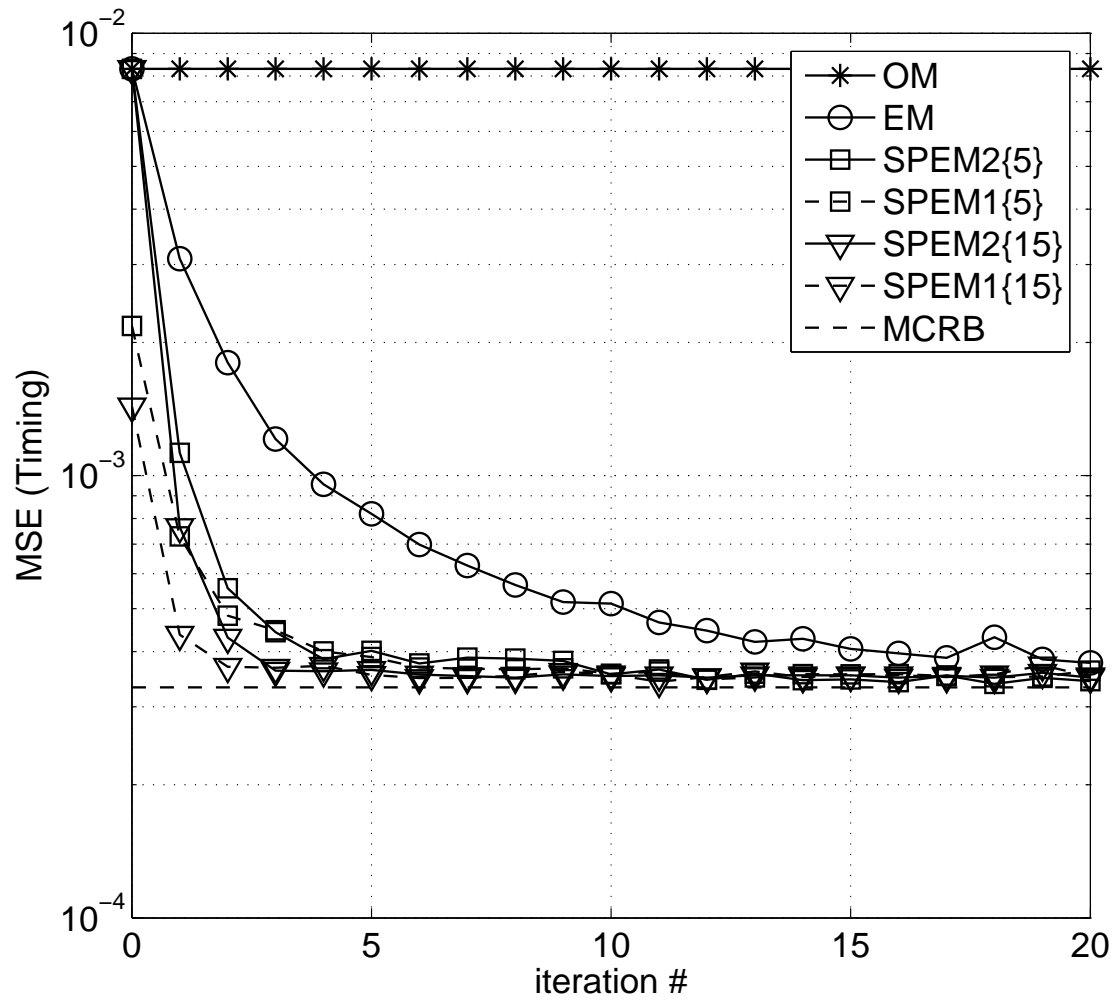


Fig. 9. MSE for timing estimation and turbo-coded transmission versus the number of turbo iterations.

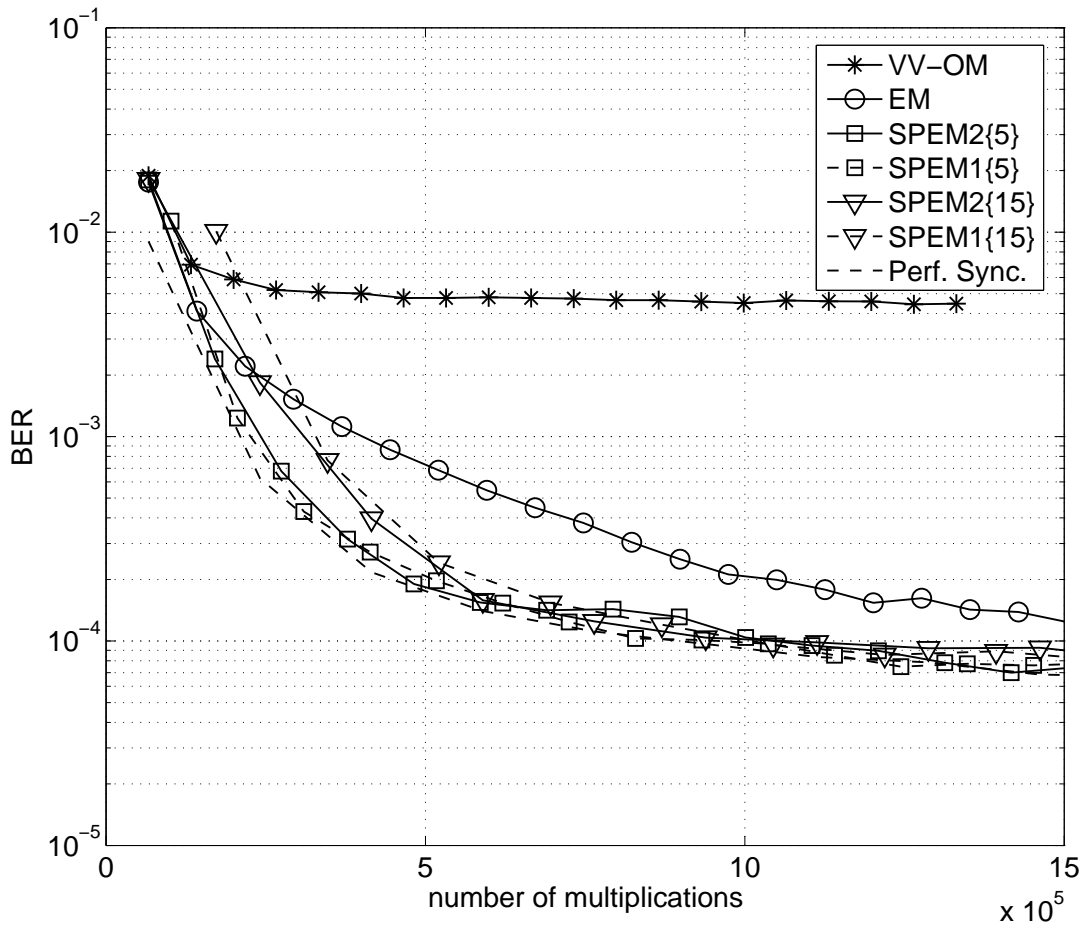


Fig. 10. BER achieved by the turbo-coded system versus the number of required multiplications.

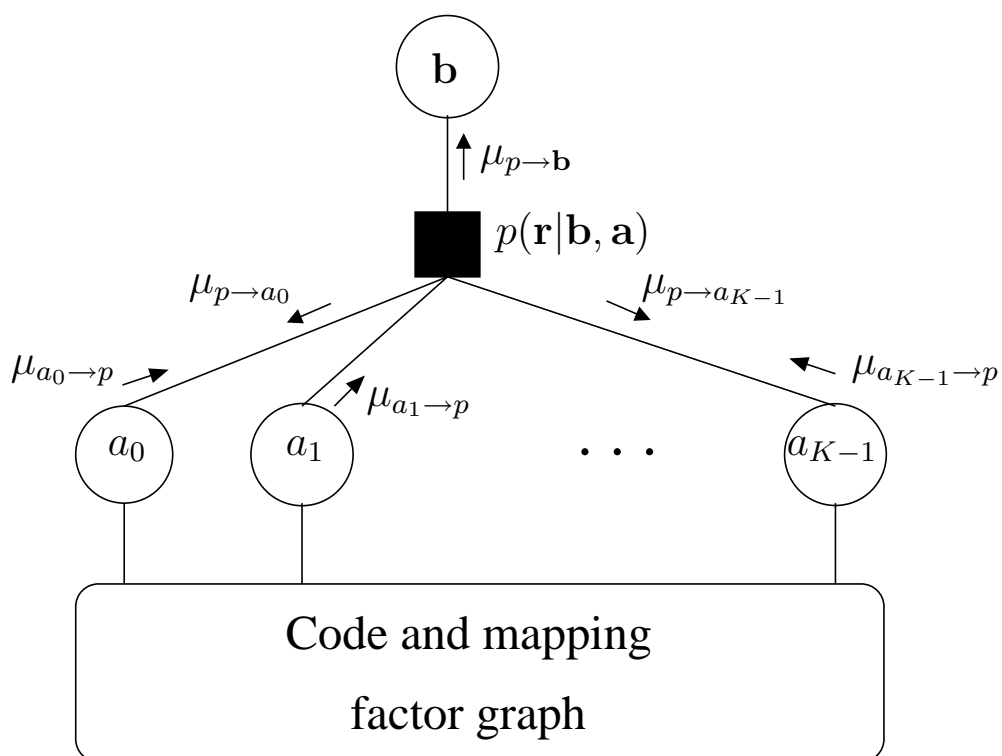


Fig. 11. Factor graph representation of  $p(\mathbf{r}, \mathbf{a} | \mathbf{b})$ . Factor nodes and variable nodes are respectively denoted by squares and circles.

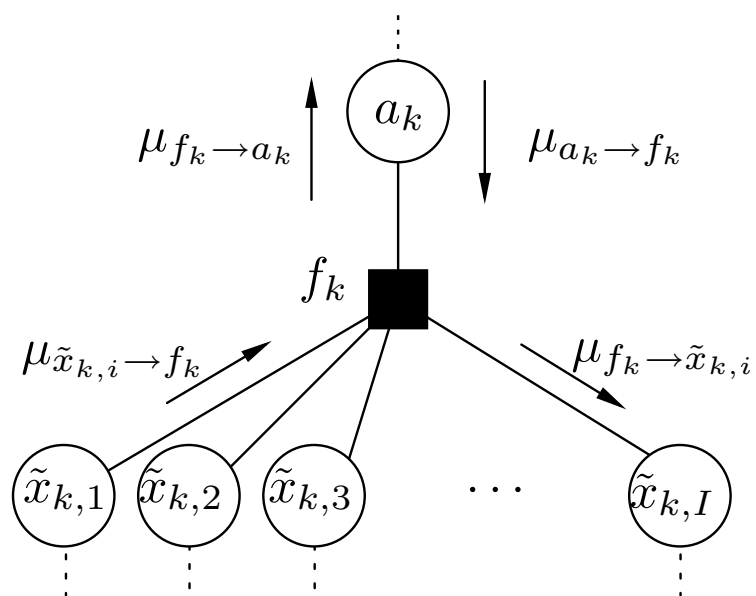


Fig. 12. Section of the factor graph of (46). Factor  $f_k$  is defined as  $f_k \triangleq \mathbb{I}\{a_k = \text{out}(\tilde{\mathbf{x}}_k)\}$ .

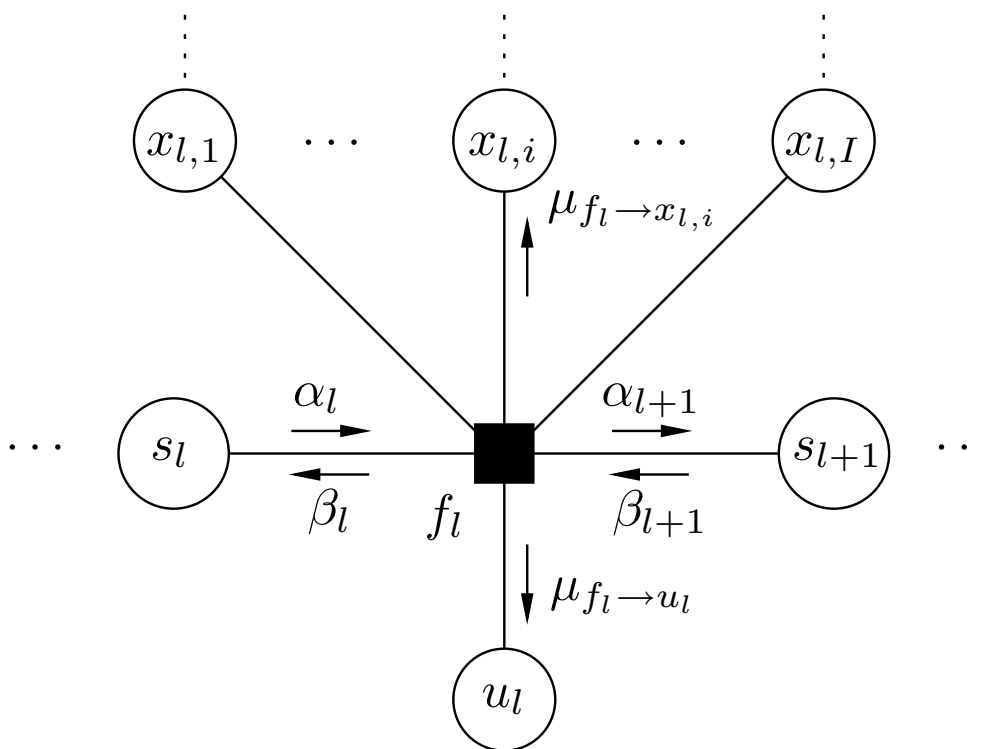


Fig. 13. Section of the factor graph of (49). Factor  $f_l$  is defined as  $f_l \triangleq \mathbb{I}\{s_{l+1} \leftarrow (u_l, s_l)\} \mathbb{I}\{\mathbf{x}_l = \text{out}(u_l, s_l)\}$ .

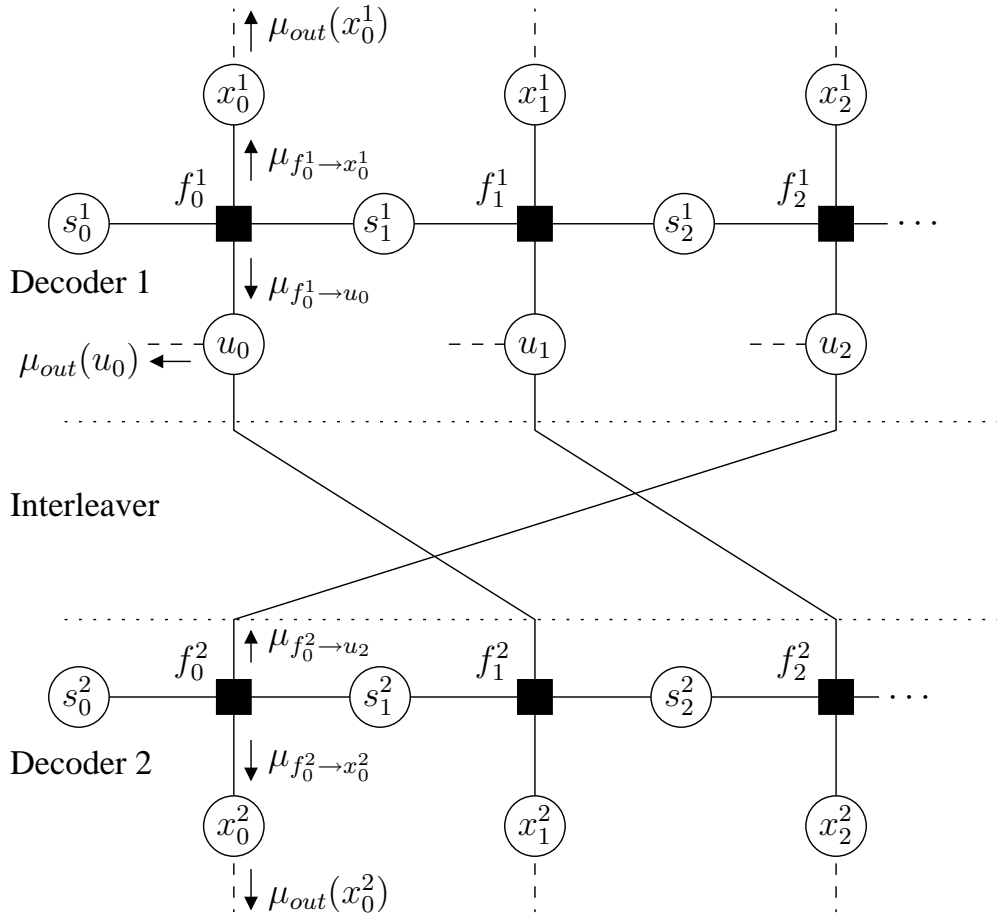


Fig. 14. Section of the factor graph relative to factorization (51). Factor  $f_l^i$  is defined as  $f_l^i \triangleq \mathbb{I}\{s_{l+1}^i \leftarrow (u_l, s_l^i)\} \mathbb{I}\{\mathbf{x}_l^i = \text{out}(u_l, s_l^i)\}$ .