INVITED PAPER

Turbo Decoding and Detection for Wireless Applications

Future iterative receivers will be designed using EXIT charts for near-capacity operation over dispersive wireless channels at the lowest possible complexity and delay.

By Lajos Hanzo, Jason P. Woodard, and Patrick Robertson

ABSTRACT | A historical perspective of turbo coding and turbo transceivers inspired by the generic turbo principles is provided, as it evolved from Shannon's visionary predictions. More specifically, we commence by discussing the turbo principles, which have been shown to be capable of performing close to Shannon's capacity limit. We continue by reviewing the classic maximum *a posteriori* probability decoder. These discussions are followed by studying the effect of a range of system parameters in a systematic fashion, in order to gauge their performance ramifications. In the second part of this treatise, we focus our attention on the family of iterative receivers designed for wireless communication systems, which were partly inspired by the invention of turbo codes. More specifically, the family of iteratively detected joint coding and modulation schemes, turbo equalization, concatenated space-time and channel coding arrangements, as well as multi-user detection and three-stage multimedia systems are highlighted.

KEYWORDS | Maximum *a posteriori* Probability (MAP) algorithm; performance of turbo codes; turbo coding; turbo equalization; turbo multiuser detection

I. INTRODUCTION

In his legendary contribution Shannon set out the performance limits of channel coding and modulation schemes as early as 1948 [1]. However, apart from Galleger's radical idea of low density parity check (LDPC) codes disseminated as early as 1962 [2], [3], no schemes were capable of approaching Shannon's capacity limits until the genesis of turbo codes in 1993 [4], [5]. It remains a mystery, why the appealing concept of exploiting the so-called *extrinsic* information provided by the surrounding bits of a bit-stream for any bits in the stream remained largely unexploited before the invention of turbo codes.

The now classic turbo-coding scheme is based on a composite codec constituted by two parallel concatenated codecs. Since their invention, turbo codes have evolved at an unprecedented pace and have reached a state of maturity within just a few years due to the intensive research efforts of the turbo coding community [6]–[10]. As a result of this dramatic evolution, turbo coding has also found its way into standardized systems, such as, for example, the recently ratified third-generation (3G) mobile radio systems [11]. Even more impressive performance gains can be attained with the aid of turbo coding in the context of video broadcast systems [12], where the associated system delay is less critical than in delay-sensitive interactive systems.

Following the above brief introduction, let us now focus our attention on a more detailed discussion of turbo coding in Section II, considering the classic turbo encoding and iterative decoding scheme, leading on to a discussion on the effects of various codec parameters on the achievable codec performance. A range of sophisticated iteratively detected wireless communications systems is proposed in Section III. More specifically, amongst a range of other schemes, turbo trellis-coded modulation is used in many of the schemes considered, leading on to turbo equalization,[1] turbo multi-user detectors (MUDs) designed

Manuscript received May 13, 2006; revised January 16, 2007. This work was supported in part by the European Commission, in part by EPSRC, U.K., and in part by the Virtual Centre of Excellence in Mobile Communications, U.K.

L. Hanzo is with the School of Electronics and Computer Science, University of Southampton, SO17 1BJ Southampton, U.K. (e-mail: lh@ecs.soton.ac.uk).

J. P. Woodard is with CSR, CB4 OWH Cambridge, U.K. (e-mail: jason.woodard@csr.com).

P. Robertson is with IKN, German Aerospace Center (DLR), 82234 Wessling, Germany (e-mail: patrick.robertson@dlr.de).

Digital Object Identifier: 10.1109/JPROC.2007.895195

[1]The term "turbo equalization" suggests that as in classic noniterative equalizers, the effects of inter-symbol interference (ISI) are mitigated. However, in recent years the employment of iterative detection has often been used in parlance synonymously with turbo equalization.

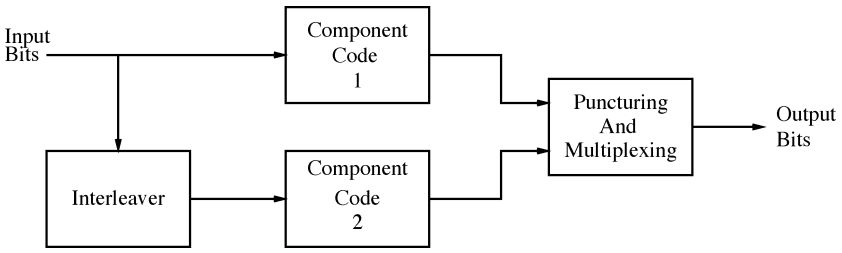

Fig. 1. ***Turbo encoder schematic, Berrou et al.*** **[4], [5].**

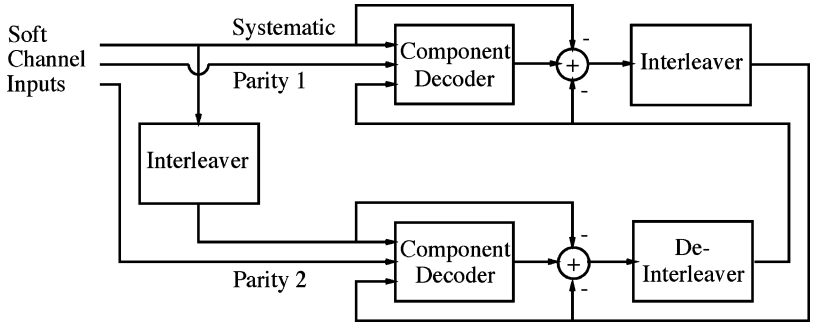

Fig. 2. ***Turbo decoder schematic.***

for both multi-user code division multiple access (CDMA), and orthogonal frequency division multiplexing (OFDM), followed by a glimpse of a three-stage iterative receiver design. Finally, we offer our conclusions in Section IV.

II. TURBO ENCODING AND DECODING

A. Turbo Encoder and Decoder Structure

The general structure used in turbo encoders is shown in Fig. 1. Two component codes are used for encoding the same input bits, but an interleaver is placed between the encoders. Generally, recursive systematic convolutional (RSC) codes are used as the component codes, but it is possible to achieve good performance using a structure like that seen in Fig. 1 with the aid of other component codes, such as for example block codes, as advocated by Hagenauer, Offer, and Papke [13] as well as by Pyndiah [14]. The resultant turbo Bose–Chaudhuri–Hocquenghem (BCH) codes have been characterized in substantial detail in [9]. Furthermore, it is also possible to employ different-rate constituent codes [9] as well as more than two component codes. However, in this treatise we will initially concentrate on the classic turbo encoder structure using two RSC codes.

Let us continue our discourse by considering the general structure of the iterative turbo decoder shown in Fig. 2. Two component decoders are linked by interleavers in a structure similar to that of the encoder. As seen in the figure, each decoder takes three inputs: the systematically encoded channel output bits, the parity bits transmitted from the associated component encoder, and the information from the other component decoder about the likely values of the bits concerned. This information from the other decoder is referred to as *a priori* information. The component decoders have to exploit both the inputs from the channel and this *a priori* information. They must also provide what are known as soft outputs for the decoded bits. This means that as well as providing the decoded output bit sequence, the component decoders must also give the associated probabilities for each bit that it has been correctly decoded. Two suitable decoders are the so-called soft output Viterbi algorithm (SOVA) proposed by Hagenauer and Hoeher [15], and the Maximum *a posteriori* Probability (MAP) [16] algorithm of Bahl, Cocke, Jelinek and Raviv, which is hence often referred to as the BCJR algorithm.[2] The MAP algorithm will be detailed in Section II-B [9].

The soft outputs from the component decoders are typically represented in terms of the so-called log likelihood ratios (LLRs), the polarity of which gives the sign of the bit, and the amplitude the probability of a

[2]In the literature, the BCJR decoder is often referred to as the MAP decoder, although more precisely it is the MAP decoder *per symbol*, while the VA may be viewed as the MAP decoder *per sequence*.

correct decision. The LLRs are simply, as their name implies, the logarithm of the ratio of two probabilities. For example, the LLR $L(u_k)$ for the value of a decoded bit u_k is given by

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

where $P(u_k = +1)$ is the probability that the bit $u_k = +1$, and similarly for $P(u_k = -1)$. Notice that the two possible values of the bit u_k are taken to be $+1$ and -1, rather than one and zero, as this simplifies the derivations that follow.

The decoder of Fig. 2 operates iteratively, and in the first iteration the first component decoder takes channel output values only and produces a soft output as its estimate of the data bits. The soft output from the first encoder is then used as additional information for the second decoder, which uses this information along with the channel outputs to calculate its estimate of the data bits. Now the second iteration can begin, and the first decoder decodes the channel outputs again, but now with additional information about the value of the input bits provided by the output of the second decoder in the first iteration. This additional information allows the first decoder to obtain a more accurate set of soft outputs, which are then used by the second decoder as *a priori* information. This cycle is repeated and upon every further iteration the bit-error rate (BER) tends to decrease. However, typically a gradually diminishing incremental BER reduction is attained. Hence, a tradeoff between the complexity imposed and the BER attained must be struck.

Due to the interleaving used at the encoder, care must be taken to properly interleave and de-interleave the LLRs which are used to represent the soft values of the bits, as seen in Fig. 2. Furthermore, because of the iterative nature of the decoding, care must be taken not to reuse the same information more than once at each decoding step. For this reason, the concepts of so-called *extrinsic* and *intrinsic* information are essential when describing the iterative decoding of turbo codes [4]. These concepts, including the reason for using the subtraction circles shown in Fig. 2, are described in Section II-C. Having considered the basic decoder structure, let us now focus our attention on the MAP algorithm in the next section.

B. Maximum *A posteriori* Algorithm

1) Introduction and Mathematical Preliminaries: In 1974, an algorithm, which has become known as the Maximum *A posteriori* (MAP) algorithm, was proposed by Bahl *et al.* [16] in order to estimate the *a posteriori* probabilities of the states and the transitions of a Markov source observed in memoryless noise. Bahl *et al.* showed how the algorithm could be used to decode both block and convolutional codes. When used to decode convolutional codes, the algorithm is optimal in terms of minimizing the decoded BER, unlike the Viterbi algorithm (VA) [17], [18], which minimizes the probability of selecting an incorrect path through the trellis. Nevertheless, as stated by Bahl *et al.* in [16], in most applications the BER performance of the two algorithms will be almost identical. However, the MAP algorithm implicitly takes into consideration every possible path through the convolutional decoder's trellis and is somewhat more complex than the VA. Hence, the MAP decoder was not widely used before the discovery of turbo codes, which required a soft-value for each bit.

The fact that the MAP algorithm provides the *a posteriori* probabilities for each bit is essential for the iterative decoding of turbo codes proposed by Berrou *et al.* [4], and so MAP decoding was used in this seminal paper. Since then, research efforts have been invested in reducing the complexity of the MAP algorithm to a reasonable level. In this section, we describe the theory behind the MAP algorithm as used for the soft output decoding of the component convolutional codes of turbo codes. It is assumed that binary codes are used.

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

$$L(u_k|\underline{y}) = \ln\left(\frac{P(u_k = +1|\underline{y})}{P(u_k = -1|\underline{y})}\right). \tag{2}$$

If the previous state $S_{k-1} = s'$ and the present state $S_k = s$ are known in a trellis, then the input bit u_k, which triggered the transition between these states becomes known. This, along with Bayes' rule and the fact that the transitions between the previous state S_{k-1} and the present state S_k in a trellis are mutually exclusive (i.e., only one of them could have occurred at the encoder), allow us to rewrite (2) as

$$L(u_k|\underline{y}) = \ln\left(\frac{\sum_{\substack{(s',s)\Rightarrow \\ u_k=+1}} p(S_{k-1} = s' \wedge S_k = s \wedge \underline{y})}{\sum_{\substack{(s',s)\Rightarrow \\ u_k=-1}} p(S_{k-1} = s' \wedge S_k = s \wedge \underline{y})}\right) \tag{3}$$

where $(s', s) \Rightarrow u_k = +1$ is the set of transitions from the previous state $S_{k-1} = s'$ to the present state $S_k = s$ that can occur if the input bit $u_k = +1$ and similarly for

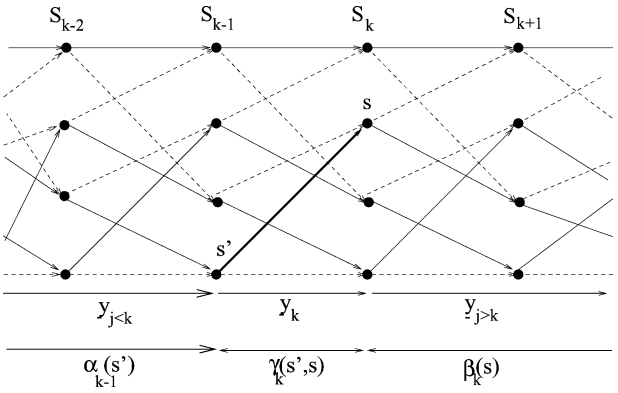

Fig. 3. ***MAP decoder trellis for K = 3 RSC code.***

$(s', s) \Rightarrow u_k = -1$. For brevity, we shall write $p(S_{k-1} = s' \wedge S_k = s \wedge \underline{y})$ as $p(s' \wedge s \wedge \underline{y})$.

We now consider the individual joint densities $p(s' \wedge s \wedge \underline{y})$ from the numerator and denominator of (3). The received sequence $\underline{y}$ can be split into three sections: the received value associated with the present transition $\underline{y}_k$, the received sequence prior to the present transition $\underline{y}_{j<k}$, and the received sequence after the present transition $\underline{y}_{j>k}$. We can thus write for the individual joint densities $p(s' \wedge s \wedge \underline{y})$

$$p(s' \wedge s \wedge \underline{y}) = p(s' \wedge s \wedge \underline{y}_{j<k} \wedge \underline{y}_k \wedge \underline{y}_{j>k}). \tag{4}$$

Fig. 3, which depicts a section of a four-state trellis for an RSC code having a constraint-length of $K = 3$, shows this split of the received channel output sequence. In this figure, solid lines represent transitions as a result of a -1 input bit, and dashed lines represent transistion resulting from a $+1$ input bit. The $\alpha_{k-1}(s')$, $\gamma_k(s', s)$, and $\beta_k(s)$ symbols represent values, which will be defined shortly, calculated by the MAP algorithm.

Bayes' rule suggests that $p(a \wedge b) = p(a|b)p(b)$. If we exploit both Bayes' rule and that the channel is memoryless, then the future received sequence $\underline{y}_{j>k}$ will depend only on the present state s, but not on the previous state s' or on the present and previous received channel output sequences $\underline{y}_k$ and $\underline{y}_{j<k}$. Hence, we can write

$$\begin{aligned} p(s' \wedge s \wedge \underline{y}) &= p(s' \wedge s \wedge \underline{y}_{j<k} \wedge \underline{y}_k \wedge \underline{y}_{j>k}) \\ &= p(\underline{y}_{j>k}|s) \cdot p(s' \wedge s \wedge \underline{y}_{j<k} \wedge \underline{y}_k) \\ &= p(\underline{y}_{j>k}|s) \cdot p\left(\{\underline{y}_k \wedge s\}|s'\right) \cdot p(s' \wedge \underline{y}_{j<k}) \\ &= \beta_k(s) \cdot \gamma_k(s', s) \cdot \alpha_{k-1}(s') \end{aligned} \tag{5}$$

where

$$\alpha_{k-1}(s') = p(S_{k-1} = s' \wedge \underline{y}_{j<k}) \tag{6}$$

is the joint probability density for the trellis at state s' at time $k-1$ and for the received channel output sequence $\underline{y}_{j<k}$, as visualized in Fig. 3. Furthermore

$$\beta_k(s) = p(\underline{y}_{j>k}|S_k = s) \tag{7}$$

is the probability density of the future received channel output sequence $\underline{y}_{j>k}$, given that the trellis is in state s at time instant k and finally

$$\gamma_k(s', s) = p\left(\{\underline{y}_k \wedge S_k = s\}|S_{k-1} = s'\right) \tag{8}$$

is the joint conditional density for the next state s and for the value $\underline{y}_k$ of the received channel output, given that the trellis was in state s' at time instant $k-1$. Equation (5) shows that the joint density $p(s' \wedge s \wedge \underline{y})$ of the encoder's trellis traversing from state $S_{k-1} = s'$ to state $S_k = s$ and that of encountering the sequence $\underline{y}$ can be represented as the product of $\alpha_{k-1}(s')$, $\gamma_k(s', s)$, and $\beta_k(s)$. The interpretation of these three probability density terms is shown in Fig. 3, where the transition $S_{k-1} = s'$ to $S_k = s$ is shown by the bold line. From (3) and (5), we can write for the conditional LLR of u_k, given the received value $\underline{y}_k$

$$\begin{aligned} L(u_k|\underline{y}) &= \ln\left(\frac{\sum_{\substack{(s',s)\Rightarrow \\ u_k=+1}} p(S_{k-1} = s' \wedge S_k = s \wedge \underline{y})}{\sum_{\substack{(s',s)\Rightarrow \\ u_k=-1}} p(S_{k-1} = s' \wedge S_k = s \wedge \underline{y})}\right) \\ &= \ln\left(\frac{\sum_{\substack{(s',s)\Rightarrow \\ u_k=+1}} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}{\sum_{\substack{(s',s)\Rightarrow \\ u_k=-1}} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}\right). \end{aligned} \tag{9}$$

The MAP algorithm finds $\alpha_k(s)$ and $\beta_k(s)$ for all states s throughout the trellis, i.e., for $k = 0, 1 \cdots N-1$, and $\gamma_k(s', s)$ for all possible transitions from state $S_{k-1} = s'$ to state $S_k = s$, and again for $k = 0, 1 \cdots N-1$. These values are then used in (9) to give the conditional LLRs $L(u_k|\underline{y})$ that the MAP decoder delivers. We now describe how the values $\alpha_k(s)$, $\beta_k(s)$, and $\gamma_k(s', s)$ can be calculated.

2) Forward Recursive Calculation of $\alpha_k(s)$: Consider first $\alpha_k(s)$. From the definition of $\alpha_{k-1}(s')$ in (6), we can write

$$\begin{aligned}\alpha_k(s) &= p(S_k = s \wedge \underline{y}_{j<k+1}) \\ &= p(s \wedge \underline{y}_{j<k} \wedge \underline{y}_k) \\ &= \sum_{\text{all } s'} p(s \wedge s' \wedge \underline{y}_{j<k} \wedge \underline{y}_k) \end{aligned} \tag{10}$$

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

$$\begin{aligned}\alpha_k(s) &= \sum_{\text{all } s'} p(s \wedge s' \wedge \underline{y}_{j<k} \wedge \underline{y}_k) \\ &= \sum_{\text{all } s'} p\Big(\{s \wedge \underline{y}_k\} | \{s' \wedge \underline{y}_{j<k}\}\Big) \cdot p(s' \wedge \underline{y}_{j<k}) \\ &= \sum_{\text{all } s'} p\Big(\{s \wedge \underline{y}_k\} | s'\Big) \cdot p(s' \wedge \underline{y}_{j<k}) \\ &= \sum_{\text{all } s'} \alpha_{k-1}(s') \cdot \gamma_k(s', s). \end{aligned} \tag{11}$$

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

$$\begin{aligned}\alpha_0(S_0 = 0) &= 1 \\ \alpha_0(S_0 = s) &= 0 \qquad \text{for all } \; s \neq 0. \end{aligned} \tag{12}$$

Fig. 4 shows an example of how one $\alpha_k(s)$ value, for $s = 0$, is calculated recursively using values of $\alpha_{k-1}(s')$ and $\gamma_k(s', s)$ for our $K = 3$ RSC code. Notice that, as we are considering a binary trellis, only two previous states, $S_{k=1} = 0$ and $S_{k-1} = 1$, have paths to the state $S_k = 0$. Therefore, the summation in (11) is over only two terms.

3) Backward Recursive Calculation of $\beta_k(s)$: The values of $\beta_k(s)$ can similarly be calculated recursively. Using a similar derivation to that for (11), it can be shown that

$$\beta_{k-1}(s') = p(\underline{y}_{j>k-1} | s') = \sum_{\text{all } s} \beta_k(s) \cdot \gamma_k(s', s). \tag{13}$$

Thus, once the values $\gamma_k(s', s)$ are known, a backward recursion can be used to calculate the values of $\beta_{k-1}(s')$ from the values of $\beta_k(s)$ using (13). Fig. 4 again shows an example of how the $\beta_k(0)$ value is calculated recursively using values of $\beta_{k+1}(s)$ and $\gamma_{k+1}(0, s)$ for our $K = 3$ RSC code.

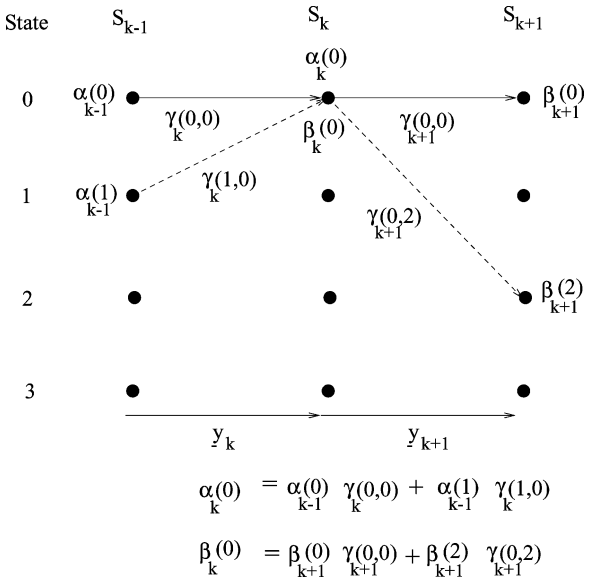

Fig. 4. ***Recursive calculation of*** $\alpha_k(0)$ ***and*** $\beta_k(0)$.

4) Calculation of $\gamma_k(s', s)$: We now consider how the transition densities $\gamma_k(s', s)$ in (5) can be calculated from the received channel sequence and any *a priori* information that is available. Using the definition of $\gamma_k(s', s)$ from (8) and the derivation from Bayes' rule we have

$$\begin{aligned}\gamma_k(s', s) &= p\Big(\{\underline{y}_k \wedge s\} | s'\Big) = p\Big(\underline{y}_k | \{s' \wedge s\}\Big) \cdot P(s | s') \\ &= p\Big(\underline{y}_k | \{s' \wedge s\}\Big) \cdot P(u_k) \\ &= p(\underline{y}_k | \underline{x}_k) \cdot P(u_k) \end{aligned} \tag{14}$$

where u_k is the input bit necessary to cause the transition from state $S_{k-1} = s'$ to state $S_k = s$, $P(u_k)$ is the *a priori* probability of this bit, and $\underline{x}_k$ is the transmitted codeword associated with this transition. Hence, the transition probability density $\gamma_k(s', s)$ is given by the product of the *a priori* probability of the input bit u_k necessary for the transition and the conditional density of the received channel sequence for the value $\underline{y}_k$ given that the codeword $\underline{x}_k$ associated with the transition was transmitted. The *a priori* probability $P(u_k)$ is derived in an iterative decoder from the output of the previous component decoder, and—assuming a memoryless additive white Gaussian noise (AWGN) channel and BPSK

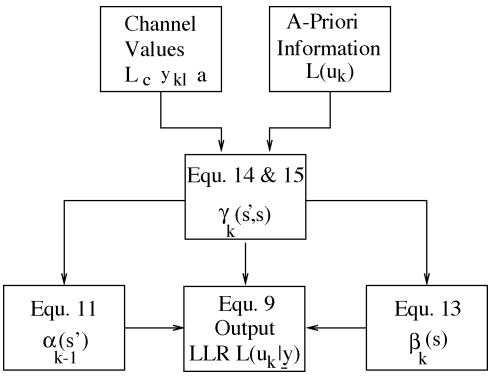

Fig. 5. ***Summary of key operations in MAP algorithm.***

modulation—the conditional received sequence probability density $p(\underline{y}_k|\underline{x}_k)$ is given by

$$p(\underline{y}_k|\underline{x}_k) = \prod_{l=1}^{n} p(y_{kl}|x_{kl}) = \prod_{l=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{\left(-\frac{E_b R}{2\sigma^2}(y_{kl}-ax_{kl})^2\right)} \tag{15}$$

where x_{kl} and y_{kl} are the individual bits within the transmitted and received codewords $\underline{y}_k$ and $\underline{x}_k$, n is the number of these bits in each codeword, E_b is the transmitted energy per bit, σ^2 is the noise variance, and a is the channel gain and we have $a = 1$ for the nonfading AWGN channel used in this paper. When considering fading channels, the value of a represents the fading magnitude.

5) Summary of MAP Algorithm: From the description given above, we see that the MAP decoding of a received sequence $\underline{y}$ to give the *a posteriori* LLR $L(u_k|\underline{y})$ can be carried out as follows. As the channel values y_{kl} are received, they and the *a priori* LLRs $L(u_k)$,[3] which are provided in an iterative turbo decoder by the other component decoder as described in Section II-C, are used to calculate $\gamma_k(s',s)$ according to (14) and (15). As the channel values y_{kl} are received and the $\gamma_k(s',s)$ values are calculated, the forward recursion from (11) can be used to calculate $\alpha_k(s',s)$. Once all the channel values have been received, and $\gamma_k(s',s)$ has been calculated for all $k = 1, 2 \cdots N$, the backward recursion from (13) can be used to calculate the $\beta_k(s',s)$ values. Finally, all the calculated values of $\alpha_k(s',s)$, $\beta_k(s',s)$, and $\gamma_k(s',s)$ are used in (9) to calculate the values of $L(u_k|\underline{y})$. These operations are summarized in the flowchart of Fig. 5.

[3]Please note that in (14) the equivalent representation of $P(u_k)$ was used.

The MAP algorithm is, in the form described in this section, extremely complex due to the multiplications needed in (11) and (13) for the recursive calculation of $\alpha_k(s',s)$ and $\beta_k(s',s)$, the multiplications and exponential operations required to calculate $\gamma_k(s',s)$ using (15), and the multiplication and natural logarithm operations required to calculate $L(u_k|\underline{y})$ using (9). However, much work has been done to reduce this complexity, and the Log-MAP algorithm [19], which will be described in Section II-D, gives the same performance as the MAP algorithm, but at a significantly reduced complexity and without the numerical problems encountered by the latter. In the next section, we will first describe the principles behind the iterative decoding of turbo codes and how the MAP algorithm described in this section can be used in such a scheme, before detailing the Log-MAP algorithm.

C. Iterative Turbo Decoding Principles

1) Turbo Decoding Mathematical Preliminaries: In this section, we explain the concepts of extrinsic and intrinsic information as used by Berrou *et al.* [4] and highlight how the MAP algorithm described in the previous section, and other soft-input soft-output (SISO) decoders, can be used in the iterative decoding of turbo codes.

It can be shown [4] that, for a systematic code such as a RSC code, the output from the MAP decoder, given by (9), can be rewritten as

$$L(u_k|\underline{y}) = \ln\left(\frac{\sum_{\substack{(s',s)\Rightarrow \\ u_k=+1}} \alpha_{k-1}(s') \cdot \gamma_k(s',s) \cdot \beta_k(s)}{\sum_{\substack{(s',s)\Rightarrow \\ u_k=-1}} \alpha_{k-1}(s')\gamma_k(s',s) \cdot \beta_k(s)}\right) = L(u_k) + L_c y_{ks} + L_e(u_k) \tag{16}$$

where

$$L_e(u_k) = \ln\left(\frac{\sum_{\substack{(s',s)\Rightarrow \\ u_k=+1}} \alpha_{k-1}(s') \cdot \chi_k(s',s) \cdot \beta_k(s)}{\sum_{\substack{(s',s)\Rightarrow \\ u_k=-1}} \alpha_{k-1}(s') \cdot \chi_k(s',s) \cdot \beta_k(s)}\right). \tag{17}$$

Here, $L(u_k)$ is the *a priori* LLR given by (1), L_c is called the channel reliability measure and is given by

$$L_c = \frac{4a}{2\sigma^2} \tag{18}$$

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

$$\chi_k(s', s) = \exp\left(\frac{L_c}{2}\sum_{l=2}^{n} y_{kl}x_{yl}\right). \tag{19}$$

Thus, we can see that the *a posteriori* LLR $L(u_k|\underline{y})$ calculated with the MAP algorithm can be thought of as comprised of three terms, namely $L(u_k)$, $L_c y_{ks}$, and $L_e(u_k)$. The *a priori* LLR term $L(u_k)$ comes from $P(u_k)$ in the expression for the branch transition probability density $\gamma_k(s', s)$ in (14). This probability should come from an independent source. In most cases, we will have no independent or *a priori* knowledge of the likely value of the bit u_k and hence the *a priori* LLR $L(u_k)$ will be zero, corresponding to an *a priori* probability $P(u_k) = 0.5$. However, in the case of an iterative turbo decoder, each component decoder can provide the other decoder with an estimate of the *a priori* LLR $L(u_k)$, as described later.

The second term $L_c y_{ks}$ in (16) is the soft output of the channel for the systematic bit u_k, which was directly transmitted across the channel and received as y_{ks}. When the channel's signal-to-noise ratio (SNR) is high, the channel reliability value L_c of (18) will be high and this systematic bit will have a large influence on the *a posteriori* LLR $L(u_k|\underline{y})$. Conversely, when the channel is poor and L_c is low, the soft output of the channel for the received systematic bit y_{ks} will have less impact on the *a posteriori* LLR delivered by the MAP algorithm.

The final term in (16), $L_e(u_k)$, is derived, using the constraints imposed by the code used, from the *a priori* information sequence $L(u_n)$ and the received channel information sequence $\underline{y}$, *excluding* the received systematic bit y_{ks} and the *a priori* information $L(u_k)$ for the bit u_k. Hence, it is called the *extrinsic* LLR for the bit u_k. Equation (16) shows that the extrinsic information from a MAP decoder can be obtained by subtracting the *a priori* information $L(u_k)$ and the received systematic channel input $L_c y_{ks}$ from the soft output $L(u_k|\underline{y})$ of the decoder. This is the reason for the subtraction paths shown in Fig. 2. Equations similar to (16) can be derived for the other component decoders which are used in iterative turbo decoding.

We summarize in the following what is meant by the terms *a priori*, *a posteriori*, and *extrinsic* information, which are central concepts behind the iterative decoding of turbo codes.

a priori — The *a priori* information concerning a bit is information known before decoding commences, which accrues from a source other than the received sequence, for example from the other component decoder. It is also sometimes referred to as intrinsic information for the sake of contrasting it with the extrinsic information to be described next. For random bits initially the *a priori* information is 0.5, but after the first "half-iteration" the first component decoder provides its estimate for the second decoder.

extrinsic — The extrinsic information concerning a bit u_k is the information provided by a decoder based on both the received sequence and on the *a priori* information *excluding* both the received samples y_{ks} representing the systematic bit u_k and the *a priori* information $L(u_k)$ for the specific bit concerned, i.e., u_k. Typically, the component decoder provides this information using the constraints imposed on the transmitted sequence by the RSC code used. It processes both the received bits as well as the *a priori* information surrounding the systematic bit u_k and uses both sources of information as well as the code constraints to provide information about the value of u_k.

a posteriori — The *a posteriori* information about a bit is the information that the decoder gives taking into account *all* available sources of information about u_k. It is the *a posteriori* LLR, i.e., $L(u_k|\underline{y})$, that the MAP algorithm gives as its output.

2) Iterative Turbo Decoding: We now describe how the iterative decoding of turbo codes is carried out. Consider initially the first component decoder in the first iteration. This decoder receives the channel sequence $L_c\underline{y}^{(1)}$ containing the received versions of the transmitted systematic bits, $L_c y_{ks}$, and the parity bits, $L_c y_{kl}$, from the first encoder. Usually, to obtain a half rate code, half of these parity bits will have been punctured at the transmitter, and so the turbo decoder must insert zeros in the soft channel output $L_c y_{kl}$ for these punctured bits. The first component decoder can then process the soft channel outputs and produce its estimate $L_{11}(u_k|\underline{y})$ of the conditional LLRs of the data bits u_k, $k = 1, 2 \cdots N$. In this notation the subscript "11" in $L_{11}(u_k|\underline{y})$ indicates that this is the *a posteriori* LLR in the first iteration from the first component decoder. Note that in this first iteration the first component decoder will have no *a priori* information about the bits, and hence $P(u_k)$ in (14) giving $\gamma_k(s', s)$ will be 0.5.

Next, the second component decoder comes into operation. It receives the channel sequence $\underline{y}^{(2)}$ containing the *interleaved* version of the received systematic bits and the parity bits from the second encoder. Again, the turbo decoder will need to insert zeroes into this sequence if the parity bits generated by the encoder are punctured before transmission. However, now, in addition to the received

channel sequence $\underline{y}^{(2)}$, the decoder can use the conditional LLR $L_{11}(u_k|\underline{y})$ provided by the first component decoder to generate *a priori* LLRs $L(u_k)$ to be used by the second component decoder. Metaphorically speaking, these *a priori* LLRs $L(u_k)$, which are related to bit u_k, would be provided by an "independent conduit of information, in order to have two independent channel-impaired opinions" concerning bit u_k. This would provide a "second channel-impaired opinion" in regards to bit u_k. In an iterative turbo decoder the extrinsic information $L_e(u_k)$ from the other component decoder is used as the *a priori* LLRs, after being interleaved to arrange the decoded data bits $\underline{u}$ in the same order as they were encoded by the second encoder. The second component decoder thus uses the received channel sequence $\underline{y}^{(2)}$ and the *a priori* LLRs $L(u_k)$ (derived by interleaving the extrinsic LLRs $L_e(u_k)$ of the first component decoder) to produce its *a posteriori* LLRs $L_{12}(u_k|\underline{y})$. This is then the end of the first iteration.

For the second iteration the first component encoder again processes its received channel sequence $\underline{y}^{(1)}$, but now it also has *a priori* LLRs $L(u_k)$ provided by the extrinsic portion $L_e(u_k)$ of the *a posteriori* LLRs $L_{12}(u_k|\underline{y})$ calculated by the second component encoder; hence, it can produce an improved *a posteriori* LLR $L_{21}(u_k|\underline{y})$. The second iteration then continues with the second component decoder using the improved *a posteriori* LLRs $L_{21}(u_k|\underline{y})$ from the first encoder to derive, through (16), improved *a priori* LLRs $L(u_k)$, which it uses in conjunction with its received channel sequence $\underline{y}^{(2)}$ to calculate $L_{22}(u_k|\underline{y})$.

This iterative process continues, and with each iteration on average the BER of the decoded bits will fall. However, as seen in [20], Fig. 9, the improvement in performance for each additional iteration carried out falls as the number of iterations increases. Hence, for complexity reasons usually only around six to eight iterations are carried out, as no significant improvement in performance is obtained with a higher number of iterations.

Fig. 6 shows how the *a posteriori* LLRs $L(u_k|\underline{y})$ output from the component decoders in an iterative decoder vary with the number of iterations used. The output from the second component decoder is shown after one, two, four, and eight iterations. The input sequence consisted entirely of -1 values, hence negative *a posteriori* LLR $L(u_k|\underline{y})$ values correspond to a correct hard decision and positive values to an incorrect hard decision. The encoded bits were transmitted over an AWGN channel at a channel SNR of -1 dB and then decoded using an iterative turbo decoder using the MAP algorithm. It can be seen that as the number of iterations used increases, the number of positive *a posteriori* LLR $L(u_k|\underline{y})$ values, and hence the BER, decreases until after eight iterations there are no incorrectly decoded values. Furthermore, as the number of iterations increases, the decoders become more certain about the value of the bits and hence the magnitudes of the LLRs gradually become larger. The erroneous decisions in the figure appear in bursts, since deviating from the error-free path trellis path typically inflicts several bit errors.

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

To elaborate a little further, it is recognized that iterative decoding performs well for turbo codes despite the fact that in reality the *a priori* information used at the input of each component decoder during the consecutive iterations is not provided by a completely independent second source. The reason for this dependence is the presence of so-called undirected cycles in the *Bayesian network* associated with the turbo code. This relationship was first presented by McEliece *et al.* in [21], demonstrating that iterative turbo decoding may be viewed as a form of belief propagation—an algorithm, which is also known in many other research communities such as artificial intelligence. For belief propagation to work efficiently, the specific graph used for visualizing the route of message passing in the decoder has to have no cycles or at least has to avoid having short cycles. Even though we have subtracted the *a priori* information from the *a posteriori* LLR to be used by the other decoder at its input, this *a priori* information affects not only the specific bit-position concerned, but also nearby bit positions' outputs due to the code's memory. Hence, in the next "half-iteration" this *a priori* information will be partly spread again by the finite-duration interleaver as well as by the other constituent decoder to the very bit position from which it originated, unless the interleaver length is sufficiently high for avoiding this event, as we will demonstrate in Section II-E4.

Another justification for using the iterative arrangement described above is how well it has been found to work. In the limited experiments that have been carried out with optimal decoding of turbo codes [22]–[24], it has been found that optimal decoding performs only a fraction of a decibel (around 0.35–0.5 dB) better than iterative decoding with the MAP algorithm. Furthermore, various turbo coding schemes have been found [24], [25] that approach the Shannonian limit, which gives the best performance theoretically available, to a similar fraction of a decibel. Therefore it seems that, for a variety of codes, the iterative decoding of turbo codes gives an almost optimal performance. Hence, it is this iterative decoding structure which is almost exclusively used with turbo codes.

Having described how the MAP algorithm can be used in the iterative decoding of turbo codes, we now proceed to describe other SISO decoders, which are less

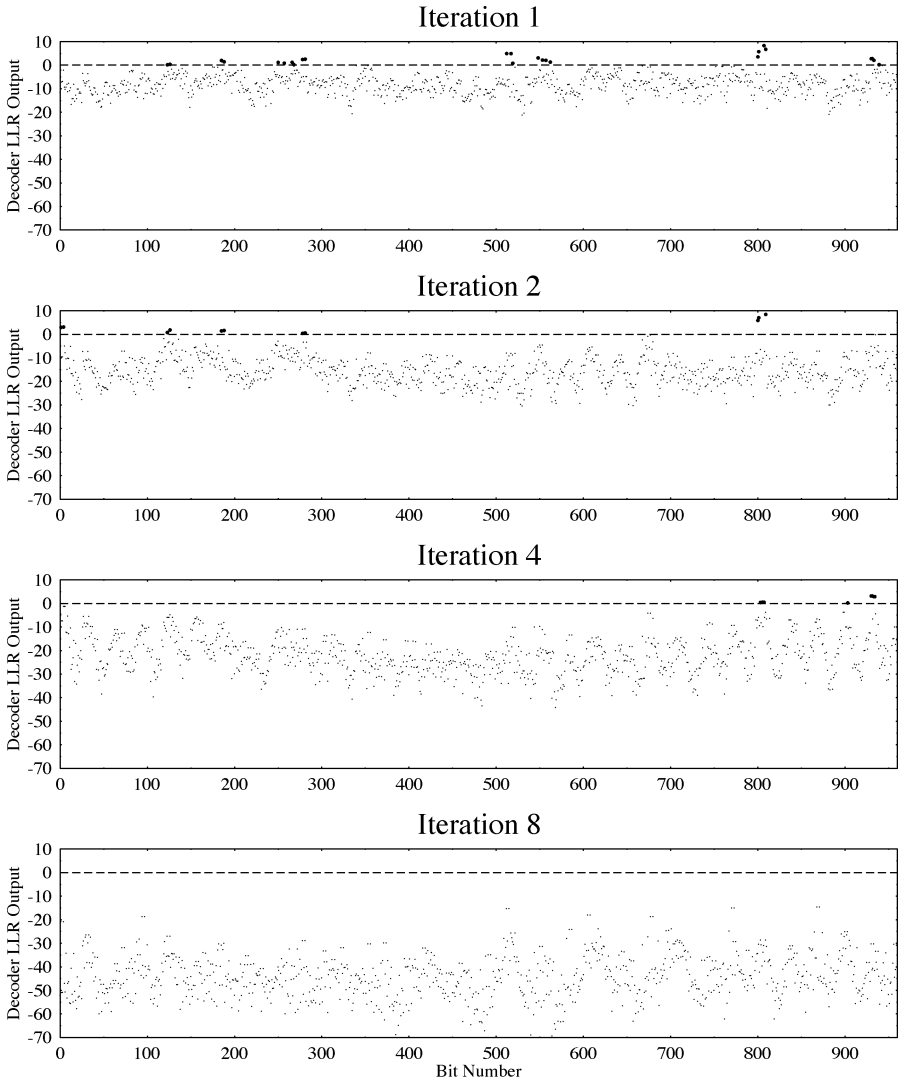

Fig. 6. ***Soft outputs from MAP decoder in iterative turbo decoder for transmitted stream of all −1.***

complex and can be used instead of the MAP algorithm. In the forthcoming section, we first describe two related algorithms, the Max-Log-MAP [26], [27] and the Log-MAP [19], which are derived from the MAP algorithm. Alternatively, the lower complexity SOVA of [9], [15], [28], and [29] may be used, which was derived from the VA.

D. Reducing the Complexity of MAP Algorithm

1) Introduction: The MAP algorithm as described in Section II-B is much more complex than the VA and with hard decision outputs performs almost identically to it. Therefore, for almost 20 years it was largely ignored. However, its application in turbo codes renewed interest in the algorithm, and it was realized that its complexity can be dramatically reduced without affecting its performance. Initially, the Max-Log-MAP algorithm was proposed by Koch and Baier [26] and Erfanian *et al.* [27]. This technique simplified the MAP algorithm by transferring the recursions into the logarithmic domain without reducing its accuracy. However, when invoking an approximation for implementing the algorithm for the sake of dramatically reducing its complexity, its performance becomes suboptimal compared to that of the MAP algorithm. As a remedy, in 1995, Robertson *et al.* [19] proposed the Log-MAP algorithm, which corrected the approximation used in the Max-Log-MAP algorithm and hence gave a performance close to that of the MAP algorithm, but at a fraction of its complexity. These two algorithms are described later in this section.

2) Max-Log-MAP Algorithm: The MAP algorithm calculates the *a posteriori* LLRs $L(u_k|\underline{y})$ using (9). To do this it requires the following values.

1) $\alpha_{k-1}(s')$ values, which are calculated in a forward recursive manner using (11);
2) $\beta_k(s)$ values, which are calculated in a backward recursion using (13);
3) branch transition densities $\gamma_k(s', s)$, which are calculated using (14).

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

$$\ln\left(\sum_i e^{x_i}\right) \approx \max_i(x_i) \tag{20}$$

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

$$A_k(s) \triangleq \ln(\alpha_k(s)) \tag{21}$$

$$B_k(s) \triangleq \ln(\beta_k(s)) \tag{22}$$

and

$$\Gamma_k(s', s) \triangleq \ln(\gamma_k(s', s)) \tag{23}$$

we can rewrite (11) as

$$\begin{aligned} A_k(s) &\triangleq \ln(\alpha_k(s)) \\ &= \ln\left(\sum_{\text{all } s'} \alpha_{k-1}(s')\gamma_k(s', s)\right) \\ &= \ln\left(\sum_{\text{all } s'} \exp[A_{k-1}(s') + \Gamma_k(s', s)]\right) \\ &\approx \max_{s'}(A_{k-1}(s') + \Gamma_k(s', s)). \end{aligned} \tag{24}$$

Equation (24) implies that for each path in Fig. 3 from the previous stage in the trellis to the state $S_k = s$ at the present stage, the algorithm adds a branch metric term $\Gamma_k(s', s)$ to the previous value $A_{k-1}(s')$ to find a new value $\tilde{A}_k(s)$ for that path. The new value of $A_k(s)$ according to (24) is then the maximum of the $\tilde{A}_k(s)$ values of the various paths reaching the state $S_k = s$. This can be thought of as selecting one path as the "survivor" and discarding any other paths reaching the state. Because of the approximation of (20) used to derive (24), only the maximum likelihood path through the state $S_k = s$ is considered when calculating this probability. Thus, the value of A_k in the Max-Log-MAP algorithm actually gives the probability of the most likely path through the trellis to the state $S_k = s$, rather than the probability of *any* path through the trellis to state $S_k = s$. This approximation is one of the reasons for the suboptimal performance of the Max-Log-MAP algorithm compared to the MAP algorithm.

We see from (24) that in the Max-Log-MAP algorithm the forward recursion used to calculate $A_k(s)$ is exactly the same as the forward recursion in the VA—for each pair of merging paths the survivor is found using two additions and one comparison. Notice that for binary trellises the summation, and maximization, over all previous states $S_{k-1} = s'$ in (24) will in fact be over only two states, because there will be only two previous states $S_{k-1} = s'$ with paths to the present state $S_k = s$. For all other values of s' we will have $\gamma_k(s', s) = 0$.

Similarly to (24), for the forward recursion used to calculate the $A_k(s)$, we can rewrite (13) as

$$\begin{aligned} B_{k-1}(s') &\triangleq \ln(\beta_{k-1}(s')) \\ &\approx \max_s(B_k(s) + \Gamma_k(s', s)) \end{aligned} \tag{25}$$

giving the backward recursion used to calculate the $B_{k-1}(s')$ values. Again, this is equivalent to the recursion used in the VA, except that it proceeds backwards rather than forwards through the trellis.

Using Equations (14) and (15), we can write the branch metrics $\Gamma_k(s', s)$ in the above recursive equations for $A_k(s)$ and $B_{k-1}(s')$ as

$$\begin{aligned} \Gamma_k(s', s) &\triangleq \ln(\gamma_k(s', s)) \\ &= C + \frac{1}{2}u_k L(u_k) + \frac{L_c}{2}\sum_{l=1}^{n} y_{kl}x_{kl} \end{aligned} \tag{26}$$

where C does not depend on u_k or on the transmitted codeword $\underline{x}_k$ and so it can be considered a constant and omitted. Hence, the branch metric is equivalent to that used in the VA, with the addition of the *a priori* LLR term $u_k L(u_k)$. Furthermore, the correlation term $\sum_{l=1}^{n} y_{kl}x_{kl}$ is weighted by the channel reliability value L_c of (18).

Finally, from (9), we can write for the *a posteriori* LLRs $L(u_k|\underline{y})$ which the Max-Log-MAP algorithm calculates

$$\begin{aligned} L(u_k|\underline{y}) &= \ln\left(\frac{\sum_{\substack{(s',s)\Rightarrow \\ u_k=+1}} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}{\sum_{\substack{(s',s)\Rightarrow \\ u_k=-1}} \alpha_{k-1}(s') \cdot \gamma_k(s', s) \cdot \beta_k(s)}\right) \\ &\approx \max_{\substack{(s',s)\Rightarrow \\ u_k=+1}}(A_{k-1}(s') + \Gamma_k(s', s) + B_k(s)) \\ &\quad - \max_{\substack{(s',s)\Rightarrow \\ u_k=-1}}(A_{k-1}(s') + \Gamma_k(s', s) + B_k(s)). \end{aligned} \tag{27}$$

This means that in the Max-Log-MAP algorithm for each bit u_k the *a posteriori* LLR $L(u_k|\underline{y})$ is calculated by considering every transition from the trellis stage S_{k-1} to the stage S_k. These transitions are grouped into those that might have occurred if $u_k = +1$ and those that might have occurred if $u_k = -1$. For both of these groups the transition giving the maximum value of $A_{k-1}(s') + \Gamma_(s', s) + B_k(s)$ is found, and the *a posteriori* LLR is calculated based on only these two "best" transitions.

The Max-Log-MAP algorithm can be summarized as follows. Forward and backward recursions—both similar to the forward recursion used in the VA—are invoked for calculating $A_k(s)$ using (24) and $B_k(s)$ employing (25). The branch metric $\Gamma_k(s', s)$ is given by (26), where the constant term C can be omitted. Once both the forward and backward recursions have been carried out, the *a posteriori* LLRs can be calculated using (27). Thus, the complexity of the Max-Log-MAP algorithm is not significantly higher than that of the VA; instead of one recursion, two are carried out, the branch metric of (26) has the additional *a priori* term $u_k L(u_k)$ term added to it, and for each bit (27) must be used to give the *a posteriori* LLRs. Viterbi states [30] that the complexity of the Log-MAP-Max algorithm may be deemed to be less than three times that of a Viterbi decoder. Unfortunately, the storage requirements are significantly higher due to the need to store both the forward and backward recursively calculated metrics $A_k(s)$ and $B_k(s)$ before the $L(u_k|\underline{y})$ values can be calculated. However, Viterbi also states [30], [31] that by increasing the computational load slightly, to four times that of the VA, the memory requirements can be dramatically reduced to become essentially equal to those of the Viterbi decoder.

3) Correcting the Approximation—Log-MAP Algorithm: The Max-Log-MAP algorithm gives a slight degradation in performance compared to the MAP algorithm due to the approximation of (20). When used for the iterative decoding of turbo codes, Robertson *et al.* [19] found this degradation to result in a drop in performance of about 0.35 dB. However, the approximation of (20) can be made exact by using the Jacobian logarithm

$$\begin{aligned}\ln(e^{x_1} + e^{x_2}) &= \max(x_1, x_2) + \ln\left(1 + e^{-|x_1 - x_2|}\right) \\ &= \max(x_1, x_2) + f_c(|x_1 - x_2|) \\ &= g(x_1, x_2) \end{aligned} \tag{28}$$

where $f_c(x)$ can be thought of as a correction term. This is then the basis of the Log-MAP algorithm proposed by Robertson, Villebrun, and Höeher [19]. Similarly to the Max-Log-MAP algorithm, values for $A_k(s) \stackrel{\Delta}{=} \ln(\alpha_k(s))$ and $B_k(s) \stackrel{\Delta}{=} \ln(\beta_k(s))$ are calculated using a forward and a backward recursion. However, the maximization in (24) and (25) is complemented by the correction term in (28). This means that the exact rather than approximate values of $A_k(s)$ and $B_k(s)$ are calculated. The correction term $f_c(\delta)$ need not be computed for every value of δ, but instead can be stored in a look-up table. Robertson *et al.* [19] found that such a look-up table need contain only eight values for δ, ranging between zero and five. This means that the Log-MAP algorithm is only slightly more complex than the Max-Log-MAP algorithm, but it gives exactly the same performance as the MAP algorithm. Therefore, it is a very attractive algorithm to use in the component decoders of an iterative turbo decoder.

Before concluding our discourse on the choice of decoding techniques, it is worth mentioning that for the specific scenario of using a simple interleaver having a few columns only it was shown in [22], [32] that it is feasible to design an optimum noniterative decoder, although its performance is limited by the employment of a suboptimum interleaver. Nonetheless, this schemes provides further insights into the so-called super-trellis structure of turbo codes and establishes their relationship with convolutional codes, since this scheme may be detected using a modified Viterbi decoder. The same philosophy was invoked also in the context of turbo equalization [33] in [34], but this topic will be detailed during our later discourse. A range of further important milestones in the development of turbo codes were created by Bendetto, Montorsi, and their team [35]–[40], which are addressed in other papers of this special issue.

Having described two techniques based on the MAP algorithm, which exhibited reduced complexity, in the next section we characterize their attainable performance.

E. Effect of Various Codec Parameters

In this section, we characterize the performance of turbo codes using binary phase shift keying (BPSK) over AWGN channels as a function of the following parameters:

1) the component decoding algorithm used;
2) the number of decoding iterations used;
3) the frame-length or latency of the input data;
4) the specific design of the interleaver used;
5) the generator polynomials and constraint-lengths of the component codes.

The standard parameters that we have used in our simulations are shown in Table 1. The turbo encoder uses two component RSCs in parallel. The RSC component codes are $K = 3$ codes with generator polynomials $G_0 = 7$ and $G_1 = 5$ in octal representation. These generator polynomials are optimum in terms of maximizing the minimum free distance of the component codes [41]. The effects of changing these parameters are examined in Section II-E5. The standard interleaver used between the two component RSC codes is a 1000-bit random interleaver with odd–even separation [42]. The effects of changing the length of the interleaver, and its structure, are examined in Section II-E4. Unless otherwise stated, the

Table 1 Standard Turbo Encoder and Decoder Parameters Used

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

results in this section are for half-rate codes, where half the parity bits generated by each of the two component RSC codes are punctured. However, for comparison, we also include some results for turbo codes where all the parity bits from both component encoders are transmitted, leading to a one-third rate code. At the decoder two SISO component decoders are used in parallel, as shown in Fig. 2. In most of our simulations we use the Log-MAP decoder, but the effect of using other component decoders is investigated in Section II-E3. Usually eight component decoder iterations are used, but in the next section we also consider the effect of different number of iterations.

1) Effect of Number of Iterations Used: Fig. 7 shows the performance of a turbo decoder using the MAP algorithm versus the number of decoding iterations which were used. For comparison, the uncoded BER and the BER obtained using convolutional coding with a standard (2,1,3) nonrecursive convolutional code are also shown. Like the component codes in the turbo encoder, the convolutional encoder uses the optimum octal generator polynomials of seven and five. It can be seen that the performance of the turbo code after one iteration is roughly similar to that of the convolutional code at low SNRs, but improves more rapidly than that of the convolutional coding as the SNR is increased. As the number of iterations used by the turbo decoder increases, the turbo decoder's performance gradually improves, although the improvements become marginal for more than eight iterations.

2) Effect of Puncturing: In our investigations we have used two RSC component encoders, and this is the arrangement most commonly used for turbo codes having code rates below $R = 2/3$. Typically, in order to give a half-rate code, half the parity bits from each component encoder are punctured. This was the arrangement used in their original paper by Berrou *et al.* on turbo codes [4]. However, it is of course possible to omit the puncturing and transmit all the parity information from both component encoders, which gives a one-third rate code. The performance of such a code, compared to the corresponding half-rate code, is shown in Fig. 8, which provides an E_b/N_0 gain of about 0.6 dB at a BER of 10^{-4}.

3) Effect of Component Decoder: Fig. 9 shows a comparison between turbo decoders using the parameters

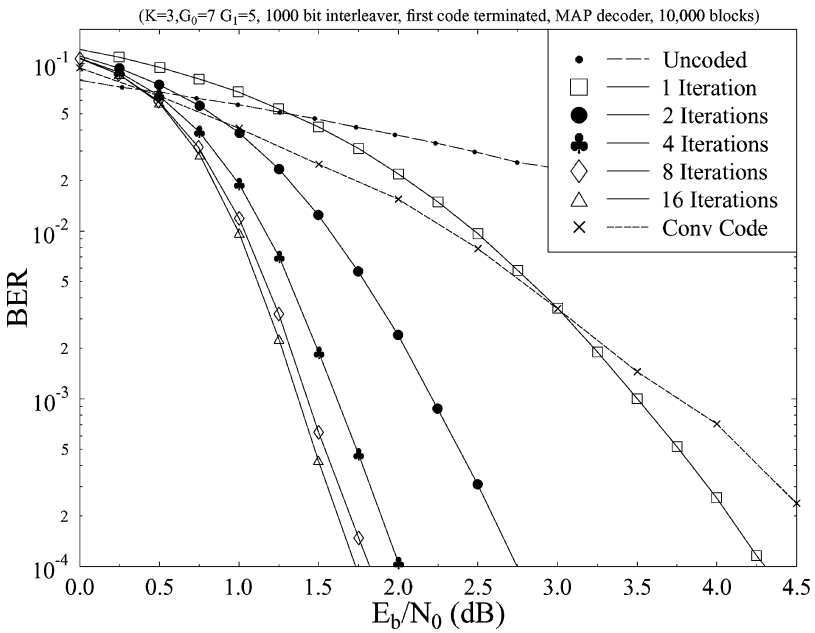

Fig. 7. ***Turbo coding BER performance using different numbers of iterations of MAP algorithm. Other parameters as in Table 1.***

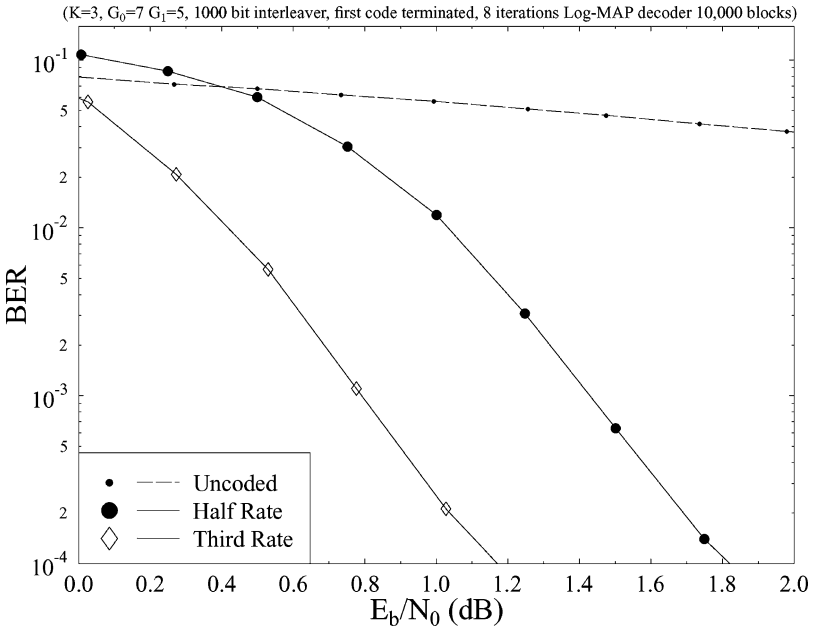

Fig. 8. *BER performance comparison between one-third and half-rate turbo codes using parameters of Table 1.*

described above. In this figure, the "Log MAP (exact)" curve refers to a decoder which calculates the correction term $f_c(x)$ in (28) of Section II-D exactly, i.e., using

$$f_c(x) = \ln(1 + e^{-x}) \tag{29}$$

rather than using a look-up table as described in [19]. The Log MAP curve refers to a decoder which does use a look-up table with eight values of $f_c(x)$ stored and hence introduces an approximation to the calculation of the LLRs. It can be seen that, as expected, the MAP and the Log-MAP (exact) algorithms give identical performances.

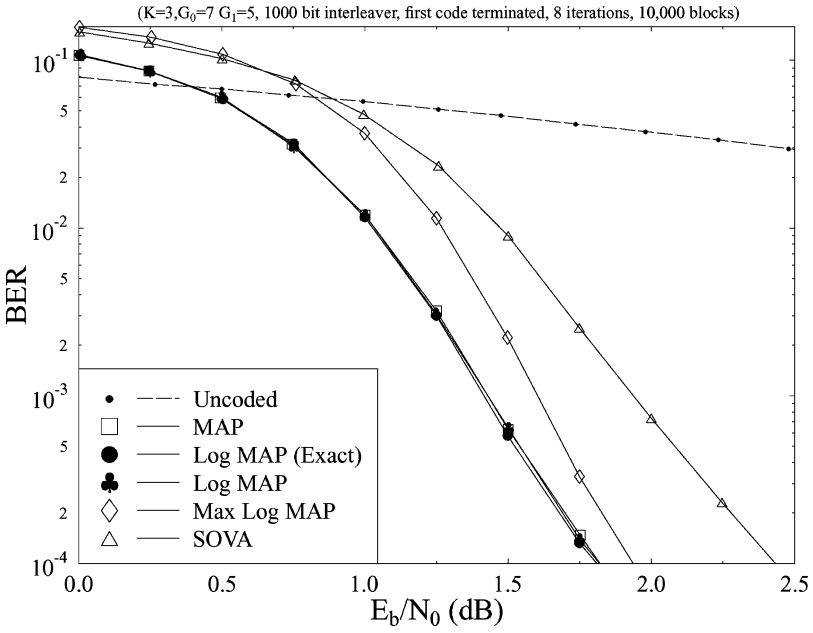

Fig. 9. *BER performance comparison between different component decoders for a random interleaver with L = 1000. Other parameters as in Table 1.*

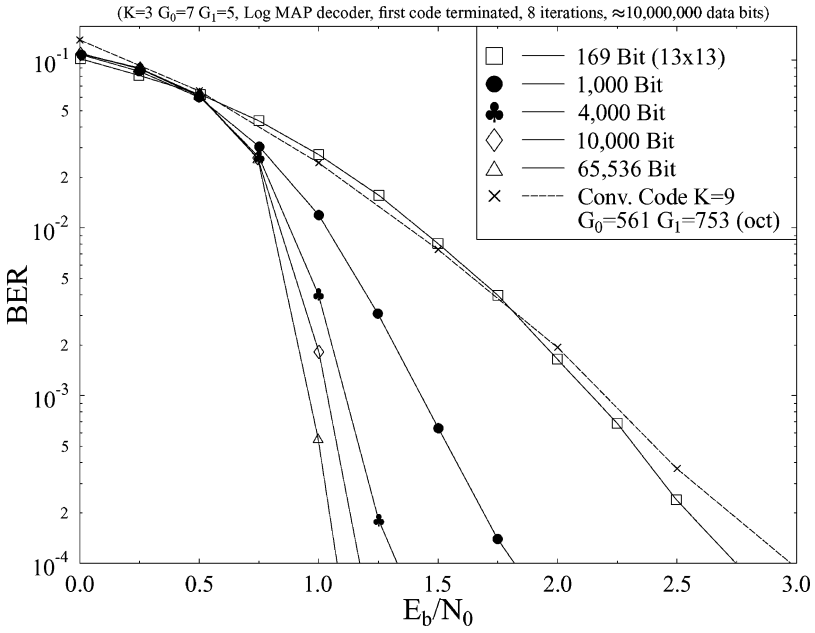

Fig. 10. ***Effect of frame length on BER performance of turbo coding. All interleavers except L = 169 block interleaver use random separated interleavers [42]. Other parameters as in Table 1.***

Furthermore, as Robertson found [19], the look-up procedure for the values of the $f_c(x)$ correction terms imposes no degradation on the performance of the decoder. It can also be seen from Fig. 9 that the Max Log MAP and the SOVA algorithms both give a degradation in performance compared to the MAP and Log MAP algorithms. At a BER of 10^{-4}, this degradation is about 0.1 dB for the Max Log MAP algorithm and about 0.6 dB for the SOVA algorithm.

4) Effect of Frame Length of Code: In the original paper on turbo coding by Berrou *et al.* [4], and many of the subsequent papers, impressive results have been presented for coding with very large frame lengths. Dolinar *et al.* analyzed the associated theoretical performance limits as a function of the coded frame length in [43] and demonstrated the benefits of turbo codes even for modest interleaver lengths.

However, for many applications, such as, for example, speech transmission systems, the large delays inherent in using high frame lengths are unacceptable. Therefore, an important area of turbo coding research is achieving as impressive results with short frame lengths as have been demonstrated for long frame-length systems. Fig. 10 shows how dramatically the performance of turbo codes depends on the frame length L used in the encoder, which is a consequence of the code's free distance being dependent on the interleaver length used. The achievable performance is comparable to or better than that of a constraint length $K = 9$ convolutional code, which has a similar complexity, as argued in [9]. In Fig. 10, a nonrecursive (2, 1, 9) convolutional code using the octal generator polynomials $G_0 = 561$ and $G_1 = 753$ was employed, which maximizes the free distance of the code [41]. These generator polynomials provide the best performance in the AWGN channels considered. A total turbo-coded frame length of 169 bits is used and the code is terminated. It can be seen that even for the short frame-length of 169 bits, turbo codes outperform similar complexity convolutional codes.

5) Effect of Component Codes: Both the constraint length and the generator polynomials used in the component codes of turbo codes are important parameters. Often, in turbo codes the generator polynomials which lead to the largest minimum free distance for ordinary convolutional codes are used, although when the effect of interleaving is considered these generator polynomials do not necessarily lead to the best minimum free distance for turbo codes. Fig. 11 shows the huge difference in performance that can result from different generator polynomials being used in the component codes. The other parameters used in these simulations were the same as detailed in Table 1.

Most of the results provided in this paper were obtained using constraint-length-3 component codes. For these codes we have used the optimum generator polynomials in terms of maximizing the minimum free distance of the component convolutional codes, i.e., seven and five in octal representation. These generator polynomials were

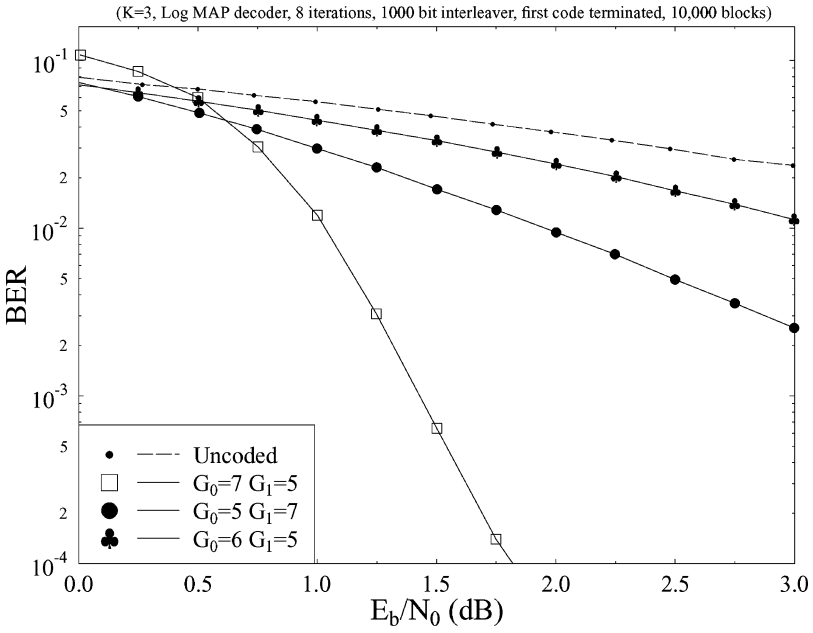

Fig. 11. ***Effect of generator polynomials on BER performance of turbo coding. Other parameters as in Table 1.***

also used for constraint-length-3 turbo coding by Hagenauer *et al.* in [13] and Jung in [44]. It can be seen from Fig. 11 that the order of these generator polynomials is important—the octal value seven should be used for the feedback generator polynomial of the encoder (denoted here by G_0). If G_0 and G_1 are swapped, the performance of a convolutional code (both regular and recursive systematic codes) would be unaffected, but for turbo codes this gives a significant degradation in performance.

The effect of increasing the constraint length of the component codes used in turbo codes is shown in Fig. 12. For the constraint-length four turbo code we again used the optimum minimum free distance generator polynomials for the component codes (15 and 17 in octal, 13 and 15 in decimal representations). The resulting turbo code gives an improvement of about 0.25 dB at a BER of 10^{-4} over the $K = 3$ curve.

For the constraint-length-5 turbo code we used the octal generator polynomials 37 and 21 (31 and 17 in decimal), which were the polynomials used by Berrou *et al.* [4] in the original paper on turbo coding. We also tried using the octal generator polynomials 23 and 35 (19 and 29), which are again the optimum minimum free distance generator polynomials for the component codes, as suggested by Hagenauer *et al.* in [13]. We found that these generator polynomials gave almost identical results to those used by Berrou *et al.*

6) Effect of Interleaver: It is well known that the interleaver used in turbo codes has a vital influence on the performance of the code. The interleaver design together with the generator polynomials used in the component codes, and the puncturing used at the encoder, have a dramatic effect on the free distance of the resultant turbo code. Several algorithms have been proposed, for example in [45] and [46], that attempt to choose good interleavers based on maximizing the minimum free distance of the code. However, this process is complex, and the resultant interleavers are not necessarily optimum. For example, in [47] random interleavers designed using the technique given in [46] are compared to a 12 × 16-dimensional block interleaver,[4] and the "optimized" interleavers are found to perform worse than the block interleaver.

In [42], a simple technique for designing good interleavers, which is referred to as "odd–even separation" is proposed. With alternate puncturing of the parity bits from each of the component codes, which is the puncturing most often used, if an interleaver is designed so that the odd and even input bits are kept separate, then it can be shown that one (and only one) parity bit associated with each information bit will be left unpunctured. This is preferable to the more general situation, where some information bits will have their parity bits from both component codes transmitted; whereas, others will have neither of their parity bits transmitted. A convenient way of achieving odd–even separation in the interleaver is to use a block interleaver with an odd

[4]A block interleaver simply writes into a rectangular memory matrix on a row-by-row basis and transmits the bits on a column-by-column basis.

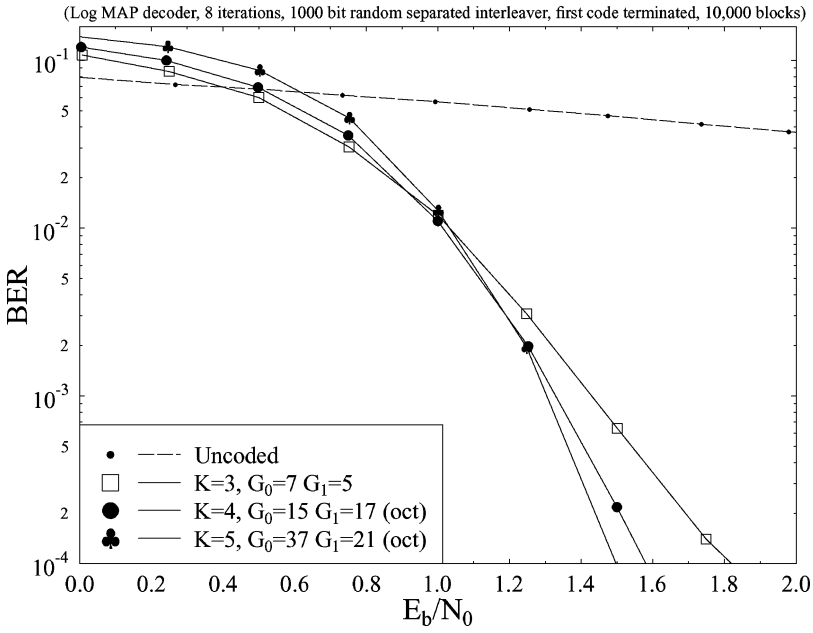

Fig. 12. ***Effect of constraint length on BER performance of turbo coding. Other parameters as in Table 1.***

number of rows and columns, which has a near-quadratic shape [42].

We also attempted using random interleavers of various frame lengths. The effect of the interleaver choice for a turbo coding system with a frame-length of approximately 960 bits is shown in Fig. 13. It can be seen from this figure that the block interleaver having an odd number of rows and columns (the 31 × 31 interleaver) performs significantly better than the interleaver with an even number of rows and columns (the 30 × 32 interleaver). However, both of these interleavers are outperformed by the two random interleavers. In the "random separated" interleaver odd–even separation, as proposed by Barbulescu and Pietrobon [42], is used. This interleaver performs very slightly better than the other random interleaver, which does not use odd–even separation. However, the effect of odd–even separation is much less significant for the random interleavers than it is for the block interleavers, but in general the employment of random interleavers is not conducive to maximizing the Hamming distance (HD). Hence, Berrou *et al.* in [48] proposed a generic model for maximizing the HD.

Having investigated the decoding algorithms as well as the achievable performance of turbo codes in conjunction with BPSK transmissions over AWGN channels, in the forthcoming section we will briefly highlight a host of related research topics motivated by the success of turbo coding. Let us commence by considering iteratively detected joint coding and multilevel modulation in the next section.

III. ITERATIVE TRANSCEIVER DESIGN FOR WIRELESS CHANNELS

A. Joint Coding and Modulation—TTCM

The invention of turbo trellis coded modulation (TTCM) by Robertson and Wöerz [49] was inspired by the joint benefits of trellis coded modulation (TCM) and turbo coding. More explicitly, incorporating the parity bits within the original bandwidth by increasing the number of bits per modulated symbol was shown to achieve a substantial coding gain, despite reducing the Eucledian distance of the modulated phasor points. Furthermore, TTCM avoids the potential disadvantage of effective throughput loss that would be incurred upon the parallel concatenation of two TCM components without invoking puncturing. Specifically, this is achieved by puncturing the parity information as proposed by Robertson and Woerz [49], so that all information bits are sent only once, and the parity bits are provided alternatively by the two component TCM encoders. The TTCM encoder mimics the structure of the turbo encoder, as seen in Fig. 14, except that it comprises two identical TCM encoders linked by a symbol interleaver, rather than two RSC encoders linked by a bit interleaver. The decoder's structure is also reminiscent of the turbo decoder, although it invokes the purely symbol-based MAP decoder detailed in [9], rather than the previously discussed bit-based MAP decoder. Since TCM was designed for AWGN channels by maximizing the Euclidean distance of the modulation constellation points, it does not perform well, when communicating over fading

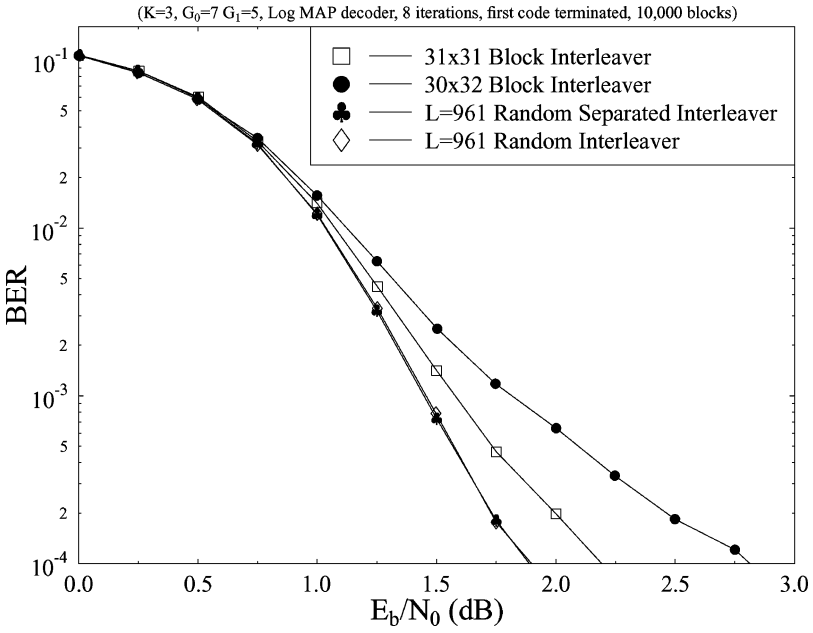

Fig. 13. *Effect of interleaver choice for L ≈ 961 frame-length turbo codes. Other parameters as in Table 1.*

channels. By contrast, bit interleaved coded modulation (BICM) [50] along with its iteratively decoded (ID) version known as BICM (BICM-ID) [51] was designed for fading channels. For an in-depth tutorial treatment and comparative performance study of the family of joint coding modulation schemes, such as TCM, TTCM, BICM, and BICM-ID, please refer to [9]. It is explicitly demonstrated in [9] that the iteratively detected TTCM scheme has a definite edge over both BICM-ID and its identical-complexity noniterative counterparts, where the complexity was quantified in terms of the total number of decoder trellis states, which in turn determines the number of add-compare-select (ACS) operations, i.e., the integrated circuit area. A range of further richly illustrated coded modulation (CM) aided examples applicable to the adaptive coding and modulation (ACM) aided high-speed downlink access (HSDPA) mode of the third-generation (3G) wireless systems and various other wireless local area networks (WLAN) can be found in [52]–[54]. Some of these CM-aided turbo transceivers will be detailed during our further discourse.

B. Turbo Equalization of Time-Variant Wireless Channels

The philosophy of turbo equalizers (TEQs) [33] is similar to that of turbo decoders, except that the extrinsic information is typically extracted from a serially concatenated, rather than from a parallel concatenated, component. More explicitly, both the channel equalizer and the channel decoder refrain from making a hard decision until they have exchanged extrinsic information in a number of iterations. The reason for this philosophy to work is,

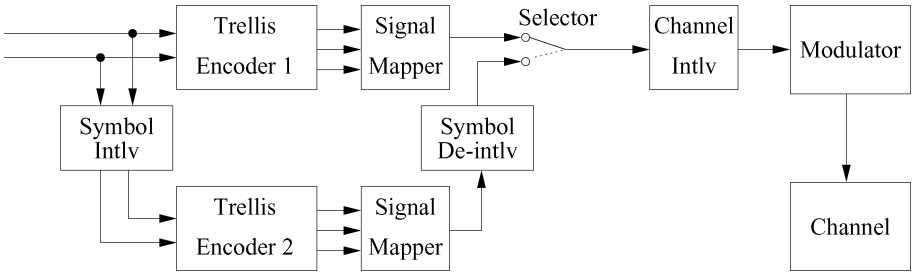

Fig. 14. *Schematic of TTCM encoder. Selector enables transmission of information bits only once and selects alternative parity bits from the constituent encoders seen at the top and bottom [49] (IEEE, 1998, Robertson and Wörz).*

because a dispersive channel has a channel impulse response (CIR), which is reminiscent of the impulse response of a convolutional encoder, the dispersive channel is often referred to as the inner encoder [9], [52]. Hence, TEQs have been shown to be successful in mitigating the effects of inter-symbol interference (ISI) introduced by both partial response modems as well as by dispersive channels and therefore they are capable of attaining a performance near that over nondispersive channels [9], [52]. Furthermore, they have the potential of mitigating the effects of channel estimation errors.

TEQs have typically been designed using a code rate of $R \leq (1/2)$. However, it was shown in [9], [52], and [55] that near-capacity performance may be attained also at higher code rates, while maintaining a high effective throughput. More explicitly, the performance of a range of BPSK turbo equalizers employing turbo BCH codes, convolutional codes, and convolutional turbo codes having high code rates, such as $R = 3/4$ and $R = 5/6$, was quantified for transmission over both a dispersive five-path Gaussian channel and an equally weighted symbol-spaced five-path Rayleigh fading channel. These turbo equalization schemes were combined with an iterative channel estimation scheme in order to characterize a realistic scenario. The results demonstrated that the turbo-equalized system using convolutional turbo codes was the most robust system for all code rates investigated, when also considering the complexity of the various arrangements. A practical turbo-equalized wireless videophone design [12] was provided in [56] in the context of the Global System of Mobile communications known as GSM.

In [57], a TEQ scheme was proposed, which employs a radial basis function (RBF)-based equalizer [52], [58], [59] instead of the classic trellis-based equalizer of Douillard *et al.* [33]. It was shown with the aid of plausible graphical examples in [52] that at the output of a dispersive fading channel the phasor constellation points may become linearly nonseparable even in the absence of noise and in this scenario only nonlinear receivers, such as the RBF-based TEQ, are capable of operating without an error floor. A novel TEQ computational complexity reduction technique was proposed, where symbol equalization was activated at any iteration, if and only if the decoded symbol had a high error probability. Otherwise, the iterations were curtailed, since a reliable decision was made. This technique provided a 37% and 54% computational complexity reduction compared to the "full-complexity" RBF TEQ for the BPSK RBF TEQ and 16QAM RBF TEQ, respectively, when communicating over dispersive Rayleigh fading channels.

In [60], 16QAM-based TCM, TTCM, BICM, and BICM-ID were amalgamated with an RBF-based TEQ scheme, which were then used as benchmarkers for a reduced complexity RBF-based TEQ scheme using separate in-phase/quadrature-phase (I/Q) TEQs. The philosophy of reduced complexity separate I/Q TEQ was first proposed in [61], where the plausible thought was capitalized on the fact that the number of channel output states to be considered is substantially reduced, if the I and Q components are considered separately. Although the following argument is somewhat simplistic, when for example 16QAM signals are transmitted over a two-path channel, $16^2 = 256$ possible phasor constellation points may be observed at the channel's output. By contrast, if we considered equalization of the quarternary I and Q components separately, we would have only $4^2 \cdot 2 = 32$ legitimate channel outputs, which results in a commensurate reduction of the number of trellis states.

To eliminate the above-mentioned flaw in this simplistic argument, when convolving the complex-valued transmitted signal with the complex-valued CIR of the channel, we are no longer at liberty to handle the I and Q components separately, since both the I and Q components contribute to both the real and imaginary channel output. Nonetheless, with the advent of the solution proposed in [61] their independent treatment is facilitated, provided that an iterative TEQ is used. To clarify this statement a little further, it is possible to estimate and compensate for the "crosstalk" between the I and Q components. In fact, even if the crosstalk were to be ignored during the first TEQ iteration, in consecutive iterations the TEQ would eliminate the crosstalk-related error at the cost of a slightly higher number of inherently less complex TEQ iterations operating on quarternary signals than that required by the 16QAM classic TEQ. This attractive complexity reduction principle is widely applicable to diverse iterative receivers and, as an example, it was extended to the reduced-complexity I/Q turbo detection of space-time trellis coded systems in [9] and [62].

Returning to [60], the least mean square (LMS) algorithm was employed for channel estimation and it was shown that both the channel estimation errors and the IQ corsstalk effects are virtually eliminated by the TEQ. As an additional benefit, the reduced-complexity RBF-I/Q-TEQ-CM achieved a similar performance to the full-complexity RBF-TEQ-CM while attaining a significant complexity reduction. The overall best performer was the RBF-I/Q-TEQ-TTCM scheme, which exhibited a coding gain of 16.78 dB. Further TEQ-related advances were reported in [63] in the context of RBF-aided space-time trellis codes designed for diversive wireless channels.

C. Concatenated FEC and Space-Time Codes

Apart from the more recent efforts of directly designing, for example, BICM [50] for employment over fading wireless channels along with BICM-ID [51], historically FEC codes have been optimized for Gaussian channels. It is a natural desire to be able to retain and exploit the vast body of knowledge on code design when communicating over fading wireless channels. This was the objective of the various studies in [9] and [10], where various space-time block codes (STBCs) and space time

trellis codes (STTCs) were concatenated with FEC codes for the sake of mitigating the effects of fading, hence virtually eliminating the effects of fading and therefore rendering the employment of designs contrived for the Gaussian channel an attractive proposition. More explicitly, the space-time codecs were concatenated with a range of channel codecs, such as convolutional and block-based turbo codes as well as with both conventional and turbo trellis-coded modulation. The associated estimated complexity issues and memory requirements were also considered: identify various space-time code, channel code combinations constituting a good engineering tradeoff in terms of their effective throughput, BER performance, and estimated complexity. It was concluded that over nondispersive fading channels the best performance versus complexity tradeoff was constituted by Alamouti's unity-rate twin-antenna block space-time code concatenated with turbo convolutional codes, while over dispersive channels space-time trellis codes had the edge. While these STTC and STBC multiple-input multiple-output (MIMO) schemes were designed for maximizing the diversity gain, the family of space division multiple access (SDMA) MIMOs [64] aims for maximizing the attainable multiplexing gain. A radical iteratively decoded variable length space time coded modulation (VL-STCM-ID) scheme capable of simultaneously providing both coding and iteration gain as well as multiplexing and diversity gain was proposed in [65], while sphere-packing modulation was employed in [66].

As a further potential system design improvement, STBC-aided inphase-quadrature phase (IQ)-interleaved TCM and TTCM schemes were proposed in [67], which are capable of quadrupling the diversity order of conventional symbol-interleaved TCM and TTCM. The increased diversity order of the proposed schemes provides significant coding gains, when communicating over nondispersive Rayleigh fading channels without compromising the coding gain achievable over Gaussian channels and without increasing the interleaving delay of the system.

D. Concatenated FEC and Iterative MUD for CDMA

The benefits of iterative wireless transceivers have also been exploited in the context of code division multiple access (CDMA) systems [68]–[70]. More specifically, a whole host of fixed-complexity channel-coded CDMA iterative parallel interference cancellation (PIC)-aided transceivers were proposed and comparatively studied in [70], which included TCM, TTCM, turbo codes, and LDPC codes. The complexity of the various schemes was quantified in terms of the total number of trellis states encountered, since this typically determines the number of ACS arithmetic operations imposed, ultimately also predetermining the required integrated circuit area. The total number of trellis states was the product of the number of states in a single decoder, the number of constituent decoders, the number of inner iterations, and the number of outer iterations. A benefit of these intelligent transceivers was that regardless of the specific choice of the FEC codec, a high coding gain and a near-single-user CDMA performance was achieved.

A radical iterative guided random search type multi-user detection (MUD) principle using genetic algorithms (GAs) was proposed in [70] and in the treatises [71]–[73], initially considering idealized synchronous, as well as more realistic asynchronous and diversity-aided CDMA systems. The appealing underlying philosophy is that in case of supporting K number of users the search-space of finding the optimum K-bit vector in the vast search space of 2^K is typically excessive, but this vector may be found with a high probability by searching only a tiny fraction of the entire search space using the GAs outlined in [70]–[73],

Nonetheless, there is a slight chance that the GA-aided MUD does not find the optimum K-bit vector, in which case an efficient FEC decoder is required for cleaning up the residual errors. Furthermore, provided that indeed, the MUD's output is protected by an FEC decoder, the search-complexity may be substantially reduced, so that the MUD's BER becomes "just" sufficiently low for the FEC decoder to clean up the residual errors. This was demonstrated in quantitative terms in the TTCM-aided system of [74], where a complexity reduction by several orders of magnitude was achieved for the specific example of $K = 10$ users and 16QAM transmission. These concepts were further developed by finding the so-called population-based soft-output GA MUD in [75].

E. Iterative Spreading-Sequence Acquisition in Multi-User CDMA

A pivotal task during the initial synchronization of mobile stations (MS) with the BS is the acquisition of the correct initial phase of the MS, before data detection may ensue. As argued previously, a TEQ is capable of elimxinating any phase error, regardless of whether it was imposed by noise, channel-induced dispersion, or CIR estimation errors. Since time-domain synchronization errors may also be deemed to impose a phase error, they can also be mitigated by appropriate iterative detectors.

In [76] and [77], a novel sequential estimation method was proposed for the acquisition of so-called m-sequences [70] that are often used as spreading sequences in CDMA systems. This sequential estimation method exploited the principle of iterative SISO decoding for enhancing the spreading sequence acquisition performance, and that of differential preprocessing for the sake of achieving an enhanced acquisition performance, when communicating in various propagation environments. Hence, the advocated acquisition arrangement was referred to as the differential recursive soft sequential estimation (DRSSE) acquisition scheme. The DRSSE acquisition scheme exhibited a low complexity, which is similar to that of an

m-sequence generator, while achieving an acquisition time that is linearly dependent on the number of delay elements in the *m*-sequence generator. A low acquisition time was maintained with the advent of determining the real-time reliabilities associated with the decision concerning a set of, say *S*, consecutive CDMA chips. This set of consecutive chips constitutes the sufficient initial condition for enabling the local *m*-sequence generator to produce a synchronized local despreading *m*-sequence replica. This technique is of particularly high importance in the context of MIMO systems, where the per-antenna transmit power is reduced while maintaining a fixed total transmit power.

F. Concatenated FEC and Iterative MUD for OFDM

Another PIC-based iterative detector designed for multiuser OFDM systems [64] was proposed in [78], which invoked combined multiuser channel estimation and iterative data detection. SDMA [64] was invoked for supporting multiple users in the uplink of the system, where the philosophy is that provided that the CIRs of the various users are sufficiently different, since they are sufficiently far apart, they can communicate within the same time-slot and frequency-slot without unduly interfering with each other. Preferably, an efficient MUD is used for separating the different users.

The proposed receiver initially estimated the channel transfer function of all users based on an approximately 5% pilot overhead, tentatively detects all users' signals and then remodulates them. The resultant signal is identical to the transmitted signal in the absence of transmission errors and hence we might argue that now a 100% pilot information is available for channel transfer function estimation. Hence, a better channel estimate is generated in the second detection iteration, which allows the system to accurately cancel the cochannel interference and make a confident final data decision. This system attains an extremely high performance.

As a further advance in the field, in [79] a GA-aided minimum mean-square error (MMSE) MUD was proposed for a TTCM assisted SDMA multiuser OFDM system, which combined the beneficial features of SDMA-OFDM [64], [78], GA-MUDs [70]–[73], and TTCM [9]. A substantial advantage of this solution was its ability to support up to a factor two higher number of users than the number of BS receiver antennas, while imposing a substantially lower complexity than the full-search-based ML detector [64], [70]. By contrast, it is widely recognized that classic MMSE-type SDMA MUDs are incapable of supporting more users than the number of BS antennas.

Another powerful technique of supporting a higher number of users than the number of antennas is constituted by the family of so-called minimum bit-error rate (MBER) receivers [80], which do not invoke the classic MMSE optimization criterion of adjusting the MUD's antenna array weights. Instead, their radical aim is that of directly minimizing the BER at the MUD's output. However, finding a solution to this optimization problem is quite a challenging task, since a number of algorithmic parameters has to be carefully adjusted, which can be iteratively carried out by GAs invoked for finding the optimum weight vectors of the MBER MUD in the context of multiple-antenna aided multi-user OFDM. In closing, we note that the iterative GA-aided MBER weight optimization is also applicable to a whole host of other wireless transceivers, as exemplified in [81] in the context of beamforming [82]. More specifically, the beamformer of [81] was capable of reducing the BER by nearly two orders of magnitude at an SNR of 10 dB in the investigated scenario in comparison to the MMSE beamforming benchmarker.

G. Iterative Detection of Three-Stage Multilevel Coding, Trellis-Coded Modulation, and Space-Time Trellis Coding

Most multimedia source signals are capable of tolerating lossy, rather than lossless, delivery to the eye, ear, and other human sensors. The corresponding lossy and preferably low-delay multimedia source codecs, however, exhibit unequal error sensitivity, which is not the case for Shannon's ideal entropy codec. In order to further advance the application of turbo detection, in [83] a jointly optimized turbo transceiver design capable of providing unequal error protection for MPEG-4 coding aided wireless video telephony [12] was proposed. The transceiver investigated consisted of STTC invoked for the sake of mitigating the effects of fading, as well as of bandwidth efficient TCM or BICM, combined with a multilevel coding (MLC) scheme employing either two different-rate nonsystematic convolutional (NSCs) codes or two RSCs for yielding a twin-class unequal-protection scheme. A single-class protection-based benchmark scheme combining STTC and NSC was used for comparison with the unequal-protection scheme advocated. The video performance of the various schemes was evaluated when communicating over uncorrelated Rayleigh fading channels. It was found that the proposed scheme required about 2.8 dB lower transmit power than the benchmark scheme in the context of the MPEG-4 videophone transceiver at a similar decoding complexity. Furthermore, the proposed twin-class STTC-TCM-RSC scheme required as low an SNR as $E_b/N_0 = 0.5$ dB in order to attain BER $= 10^{-4}$ and a video peak signal-to-noise ratio (PSNR) of 37 dB, which is 2.3 dB away from the corresponding MIMO channel's capacity [84]. However, if the proposed STTC-TCM-2RSC scheme is used for broadcasting MPEG-4 encoded video, where a longer delay can be tolerated, the required E_b/N_0 value is further reduced and it is only 1 dB away from the MIMO channel's capacity. The convergence properties of the scheme were also studied using a novel three-dimensional extrinsic information transfer chart (EXIT) [85], which was developed from the concepts of [86]–[88].

A further multistage iterative receiver was proposed in [89], while the low-complexity computation of EXIT charts for nonbinary systems was outlined in [90].

IV. CONCLUSION

The now classic developments of the turbo-coding era were briefly outlined in Section II. In order to quantify the attainable performance we also provided a range of results using a variety of codec parameters. The above-mentioned classic developments were followed by the post-turbo-coding era, leading to the invention of iterative wireless turbo receivers, which were inspired by turbo codes, as detailed in Section III. First, the benefits of turbo equalizers were discussed, which are capable of performing close to the limits derived for nondispersive channels. Furthermore, they are also capable of eliminating the effects of both channel estimation errors and synchronization errors. The joint design of diverse channel codes and space-time codes was also discussed, with the aim of achieving the highest possible coding/diversity gain at the lowest possible complexity. The turbo detection principle was then also extended to turbo multi-user detection of CDMA and OFDM systems in conjunction with a whole suite of channel codecs, again, aiming for the highest achievable coding/diversity gain at the lowest possible complexity. The family of random-guided search-based iterative soft-detection aided GA MUDs was also briefly highlighted and the benefits of multistage iterative receivers in the context of joint source and channel coding [91] as well as space-time coding were outlined.

Future iterative receiver research is expected to rely on EXIT-chart-based design principles for the sake of performing close to the capacity limits, using the principles outlined in [92], [93] and stimulating the research community to aspire for achieving near-capacity performance over dispersive, fading wireless channels at the lowest possible complexity and delay. ■

Acknowledgment

The authors would like to gratefully acknowledge the enlightenment they gained from collaboration with their valued friends and colleagues in the framework of the various joint papers cited. The insightful comments of the anonymous reviewers are also gratefully acknowledged.

REFERENCES

[1] C. E. Shannon, "A mathematical theory of communication," *Bell System Tech. J.*, pp. 379–427, 1948.

[2] R. Gallager, "Low density parity check codes," *IEEE Trans. Inform. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[3] R. Gallager, "Low density parity check codes," Ph.D. dissertation, M.I.T., Cambridge, MA, 1963.

[4] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *Proc. Int. Conf. Communications*, Geneva, Switzerland, May 1993, pp. 1064–1070.

[5] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo codes," *IEEE Trans. Commun.*, vol. 44, pp. 1261–1271, Oct. 1996.

[6] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "A soft-input soft-output APP module for iterative decoding of concatenated codes," *IEEE Commun. Lett.*, vol. 1, pp. 22–24, Jan. 1997.

[7] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Algorithm for continuous decoding of turbo codes," *IEEE Trans. Inform. Theory*, vol. 32, no. 2, pp. 314–315, Feb. 1996.

[8] S. Benedetto and G. Montorsi, "Unveiling turbo codes: Some results on parallel concatenated coding schemes," *IEEE Trans. Inform. Theory*, vol. 42, no. 3, pp. 409–428, Mar. 1996.

[9] L. Hanzo and B. Y. T. H. Liew. (2002). *Turbo Coding, Turbo Equalisation and Space-Time Coding.* New York: Wiley, IEEE Press. [Online]. Available: http://www-mobile.ecs.soton.ac.uk

[10] T. Liew and L. Hanzo, "Space-time codes and concatenated channel codes for wireless communications," *Proc. IEEE*, vol. 90, no. 2, pp. 183–219, Feb. 2002.

[11] R. Steele and L. Hanzo, Eds., *Mobile Radio Communications: Second and Third Generation Cellular and WATM Systems*, 2nd ed. New York: Wiley, 1999.

[12] L. Hanzo, P. Cherriman, and J. Streit. (2001). *Wireless Video Communications: From Second to Third Generation Systems, WLANs and Beyond.* New York: IEEE Press. [Online]. Available: http://www-mobile.ecs.soton.ac.uk

[13] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inform. Theory*, vol. 42, no. 3, pp. 429–445, Mar. 1996.

[14] R. Pyndiah, "Iterative decoding of product codes: Block turbo codes," in *Proc. Int. Symp. Turbo Codes Related Topics*, Brest, France, Sep. 1997, pp. 71–79.

[15] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," *Proc. IEEE Globecom*, pp. 1680–1686, 1989.

[16] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimising symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, no. 3, pp. 284–287, Mar. 1974.

[17] A. Viterbi, "Error bounds for convolutional codes and an asymphotically optimum decoding algorithm," *IEEE Trans. Inform. Theory*, vol. IT-13, no. 4, pp. 260–269, Apr. 1967.

[18] G. Forney, "The Viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.

[19] P. Robertson, E. Villebrun, and P. Höher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proc. Int. Conf. Communications*, Seattle, WA, Jun. 1995, pp. 1009–1013.

[20] B. Sklar, "A primer on turbo code concepts," *IEEE Commun. Mag.*, pp. 94–102, Dec. 1997.

[21] R. J. McEliece, D. J. C. MacKay, and J. F. Cheng, "Turbo decoding as an instance of Pearl's belief propagation algorithm," *IEEE J. Select. Areas in Commun.*, vol. 16, no. 2, pp. 140–152, Feb. 1998.

[22] M. Breiling and L. Hanzo, "Optimum non-iterative decoding of turbo codes," *IEEE Trans. Inform. Theory*, vol. 46, no. 9, pp. 2212–2228, Sep. 2000.

[23] M. Breiling and L. Hanzo, "Optimum non-iterative turbo-decoding," in *Proc. of PIMRC'97*, Helsinki, Finland, Sept. 1997, pp. 714–718.

[24] C. Berrou, "Some clinical aspects of turbo codes," in *Proc. Int. Symp. Turbo Codes Related Topics*, Brest, France, Sep. 1997, pp. 26–31.

[25] H. Nickl, J. Hagenauer, and F. Burkett, "Approaching Shannon's capacity limit by 0.27 dB using simple hamming codes," *IEEE Commun. Lett.*, vol. 1, no. 9, pp. 130–132, Sep. 1997.

[26] W. Koch and A. Baier, "Optimum and sub-optimum detection of coded data disturbed by time-varying inter-symbol interference," *Proc. IEEE GLOBECOM*, pp. 1679–1684, Dec. 1990.

[27] J. Erfanian, S. Pasupathy, and G. Gulak, "Reduced complexity symbol dectectors with parallel structures for ISI channels," *IEEE Trans. Commun.*, vol. 42, pp. 1661–1671, 1994.

[28] G. Battail, "Ponderation des symboles decodes par l'algorithme de viterbi (in french)," *Ann. Telecommun., France*, vol. 42, pp. 31–38, Jan. 1987.

[29] C. Berrou, P. Adde, E. Angui, and S. Faudeil, "A low complexity soft-output Viterbi decoder architecture," in *Proc. Int. Conf. Communications*, May 1993, pp. 737–740.

[30] A. Viterbi, "Approaching the Shannon limit: Theorist's dream and practitioner's challenge," in *Proc. Int. Conf. Millimeter Wave and Far Infrared Science Technology*, 1996, pp. 1–11.

[31] A. J. Viterbi, "An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes," *IEEE J. Select. Areas Commun.*, pp. 260–264, Feb. 1997.

[32] M. Breiling and L. Hanzo, "Non-iterative optimum super-trellis decoding of turbo codes," *Electron. Lett.*, vol. 33, pp. 848–849, May 1997.

[33] C. Douillard, A. Picart, M. Jézéquel, P. Didier, C. Berrou, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," *Eur. Trans. Commun.*, vol. 6, pp. 507–511, 1995.

[34] A. Knickenberg, B. Yeap, J. Hàmorskỳ, M. Breiling, and L. Hanzo, "Non-iterative joint channel equalisation and channel decoding," *Inst. Elect. Eng. Electron. Lett.*, vol. 35, pp. 1628–1630, Sep. 16, 1999.

[35] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Bandwidth efficient parallel concatenated coding schemes," *Inst. Elect. Eng. Electron. Lett.*, vol. 31, pp. 2067–2069, Nov. 23, 1995.

[36] S. Benedetto and G. Montorsi, "The role of recursive convolutional codes in turbo codes," *Electron. Lett.*, vol. 31, pp. 858–859, May 1995.

[37] S. Benedetto and G. Montorsi, "Average performance of parallel concatenated block codes," *Electron. Lett.*, vol. 31, pp. 156–158, Feb. 1995.

[38] S. Benedetto and G. Montorsi, "Performance evaluation of turbo-codes," *Electron. Lett.*, vol. 31, pp. 163–165, Feb. 1995.

[39] S. Benedetto and G. Montorsi, "Design of parallel concatenated convolutional codes," *IEEE Trans. Commun.*, vol. 44, no. 5, pp. 591–600, May 1996.

[40] S. Benedetto and G. Montorsi, "Unveiling turbo codes: Some results on parallel concatenated coding schemes," *IEEE Trans. Inform. Theory*, vol. 42, no. 3, pp. 409–428, Mar. 1996.

[41] J. Proakis, *Digital Communications*, 3rd ed. New York: McGraw-Hill, 1995.

[42] A. Barbulescu and S. Pietrobon, "Interleaver design for turbo codes," *Inst. Elect. Eng. Electron. Lett.*, pp. 2107–2108, Dec. 1994.

[43] S. Dolinar, D. Divsalar, and F. Pollara, "Code performance as a function of block size," SPL, NASA, pp. 42–133. [Online]. Available: http://tmo.jpl.nasa.gov/tmo/progress_report/42-133/title.htm

[44] P. Jung, "Comparison of turbo-code decoders applied to short frame transmission systems," *IEEE J. Select. Areas Commun.*, pp. 530–537, 1996.

[45] P. Robertson, "Illuminating the structure of code and decoder of parallel concatenated recursive systematic (turbo) codes," *Proc. IEEE GLOBECOM*, pp. 1298–1303, 1994.

[46] P. Jung and M. Nasshan, "Performance evaluation of turbo codes for short frame transmission systems," *Inst. Elect. Eng. Electron. Lett.*, vol. 30, pp. 111–112, Jan. 1994.

[47] P. Jung and M. Nasshan, "Dependence of the error performance of turbo-codes on the interleaver structure in short frame transmission systems," *Inst. Elect. Eng. Electron. Lett.*, pp. 287–288, Feb. 1994.

[48] C. Berrou, Y. Saouter, C. Douillard, S. Kerouedan, and M. Jézéquel, "Designing good permutations for turbo codes: Towards a single model," in *IEEE Int. Conf. Communications*, vol. 1, Jun. 2004.

[49] P. Robertson and T. Wörz, "Bandwidth-efficient turbo trellis-coded modulation using punctured component codes," *IEEE J. Select. Areas Commun.*, vol. 16, no. 2, pp. 206–218, Feb. 1998.

[50] E. Zehavi, "8-PSK trellis codes for a Rayleigh fading channel," *IEEE Trans. Commun.*, vol. 40, no. 5, pp. 873–883, May 1992.

[51] X. Li and J. A. Ritcey, "Bit-interleaved coded modulation with iterative decoding," *IEEE Commun. Lett.*, vol. 1, no. 11, Nov. 1997.

[52] L. Hanzo, C. Wong, and M. Yee. (2002). *Adaptive Wireless Transceivers.* New York: Wiley/IEEE. [Online]. Available: http://www-mobile.ecs.soton.ac.uk

[53] L. Hanzo, S.-X. Ng, T. Keller, and W. Webb, *Quadrature Amplitude Modulation: From Basics to Adaptive Trellis-Coded, Turbo-Equalised and Space-Time Coded OFDM, CDMA and MC-CDMA Systems.* New York: IEEE Press, 2000.

[54] R. G. Maunder, J. Kliewer, S. X. Ng, J. Wang, L.-L. Yang, and L. Hanzo, "A wireless video scheme employing the joint iterative decoding of trellis-based vector quantization and trellis-coded modulation," *IEEE Trans. Wireless Commun.*, to appear.

[55] B. L. Yeap, T. H. Liew, J. Hàmorskỳ, and L. Hanzo, "Comparative study of convolutional coded as well as convolutional-based and block-based turbo coded turbo equalisers," *IEEE Trans. Wireless Commun.*, vol. 1, no. 4, pp. 266–273, Apr. 2002.

[56] P. Cherriman, B. L. Yeap, and L. Hanzo, "The performance of H.263-based video telephony over turbo-equalised GSM/GPRS," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 10, pp. 909–916, Oct. 2002.

[57] M. S. Yee, B. L. Yeap, and L. Hanzo, "Radial basis function assisted turbo equalization," *IEEE Trans. Commun.*, vol. 51, no. 4, pp. 664–675, Apr. 2003.

[58] M. Yee, T. Liew, and L. Hanzo, "Burst-by-burst adaptive turbo coded radial basis function assisted feedback equalisation," *IEEE Trans. Commun.*, vol. 49, no. 11, Nov. 2001.

[59] M. Yee and L. Hanzo, "A wideband radial basis function decision feedback equaliser assisted burst-by-burst adaptive modem," *IEEE Trans. Commun.*, vol. 50, no. 5, pp. 693–697, May 2002.

[60] S. X. Ng, M. S. Yee, and L. Hanzo, "Coded modulation assisted radial basis function aided turbo equalisation for dispersive Rayleigh fading channels," *IEEE Trans. Wireless Commun.*, vol. 3, no. 11, pp. 2198–2206, Nov. 2004.

[61] B. L. Yeap, C. H. Wong, and L. Hanzo, "Reduced complexity in-phase/quadrature-phase m-QAM turbo equalization using iterative channel estimation," *IEEE Trans. Wireless Commun.*, vol. 2, no. 1, pp. 2–10, 2002.

[62] B. Yeap and L. Hanzo, "Reduced complexity I/Q turbo detection for space-time trellis coded systems," *IEEE Trans. Veh. Technol.*, vol. 53, no. 7, pp. 1278–1286, Jul. 2004.

[63] M.-S. Yee, B.-L. Yeap, and L. Hanzo, "RBF-based decision feedback aided turbo equalisation of convolutional and space-time trellis coded systems," *Inst. Elect. Eng. Electron. Lett.*, vol. 37, pp. 1298–1299, Oct. 11, 2001.

[64] L. Hanzo, M. Münster, B. Choi, and T. Keller. (2003). *OFDM and MC-CDMA for Broadband Multiuser Communications, WLANs and Broadcasting.* New York: Wiley/IEEE. [Online]. Available: http://www-mobile.ecs.soton.ac.uk

[65] S. X. Ng, J. Wang, M. Tao, L.-L. Yang, and L. Hanzo, "Iteratively decoded variable-length space-time coded modulation: Code construction and convergence analysis," *IEEE Trans. Wireless Commun.*, 2006, to appear.

[66] O. Alamri, B. L. Yeap, and L. Hanzo, "A turbo detection and sphere packing modulation aided space-time coding scheme," *IEEE Trans. Veh. Technol.*, Feb. 2007.

[67] S. X. Ng and L. Hanzo, "Space-time IQ-interleaved TCM and TTCM for AWGN and Rayleigh fading channels," *Inst. Elect. Eng. Electron. Lett.*, vol. 38, pp. 1553–1555, Nov. 21, 2002.

[68] X. Wang and H. Poor, "Blind equalization and multiuser detection in dispersive CDMA channels," *IEEE Trans. Commun.*, vol. 46, no. 1, pp. 91–103, Jan. 1998.

[69] X. Wang and H. V. Poor, "Adaptive joint multiuser detection and channel estimation in multipath fading CDMA," *Wireless Networks*, vol. 4, pp. 453–470, Jun. 1998.

[70] L. Hanzo, L.-L. Yang, E.-L. Kuan, and K. Yen. (2003). *Single- and Multi-Carrier DS-CDMA.* New York: Wiley/IEEE. [Online]. Available: http://www-mobile.ecs.soton.ac.uk

[71] K. Yen and L. Hanzo, "Genetic algorithm assisted joint multiuser symbol detection and fading channel estimation for synchronous CDMA systems," *IEEE J. Select. Areas Commun.*, vol. 19, no. 1, pp. 985–998, Jun. 2001.

[72] K. Yen and L. Hanzo, "Antenna diversity assisted genetic algorithm based multiuser detection schemes for synchronous CDMA systems," *IEEE Trans. Commun.*, vol. 51, no. 3, pp. 366–370, Mar. 2003.

[73] K. Yen and L. Hanzo, "Genetic algorithm assisted multiuser detection in asynchronous CDMA communications," *IEEE Trans. Veh. Technol.*, vol. 53, no. 9, pp. 1413–1422, Sep. 2004.

[74] S. X. Ng, K. Yen, and L. Hanzo, "TTCM assisted genetic-algorithm aided reduced-complexity multiuser detection," *Inst. Elect. Eng. Electron. Lett.*, vol. 38, pp. 722–724, Jul. 4, 2002.

[75] M. Jiang, J. Akhtman, and L. Hanzo, "Soft-information assisted near-optimum nonlinear detection for BLAST-type space division multiplexing OFDM systems," *IEEE Trans. Wireless Commun.*, vol. 6, no. 3, Mar. 2007.

[76] L.-L. Yang and L. Hanzo, "Iterative soft sequential estimation assisted acquisition of *m*-sequences," *Inst. Elect. Eng. Electron. Lett.*, vol. 38, pp. 1550–1551, Nov. 21, 2002.

[77] L.-L. Yang and L. Hanzo, "Differential acquisition of *m*-sequences using recursive soft sequential estimation," *IEEE Trans. Wireless Commun.*, vol. 4, no. 1, pp. 128–136, Jan. 2005.

[78] M. Münster and L. Hanzo, "Parallel interference cancellation assisted decision-directed channel estimation for OFDM systems using multiple transmit antennas," *IEEE Trans. Wireless Commun.*, vol. 4, pp. 2148–2162, Sep. 2005.

[79] M. Jiang and L. Hanzo, "Improved hybrid MMSE detection for turbo trellis coded modulation assisted multi-user OFDM systems," *Inst. Elect. Eng. Electron. Lett.*, vol. 40, no. 16, pp. 1002–1003, 2004.

[80] M. Alias, S. Chen, and L. Hanzo, "Multiple-antenna-aided OFDM employing genetic-algorithm-assisted minimum bit error rate multiuser detection," *IEEE Trans. Veh. Technol.*, vol. 54, Sep. 2005.

[81] A. Wolfgang, N. Ahmad, S. Chen, and L. Hanzo, "Genetic algorithm assisted error probability optimisation for beamforming," *Inst. Elect. Eng. Electron. Lett.*, vol. 40, no. 5, pp. 320–322, 2004.

[82] J. Blogh and L. Hanzo. (2002). *3G Systems and Intelligent Networking*. New York: Wiley/IEEE. [Online]. Available: http://www-mobile.ecs.soton.ac.uk

[83] S. X. Ng, J. Y. Chung, and L. Hanzo, "Turbo-detected unequal protection MPEG-4 wireless video telephony using multi-level coding, trellis coded modulation and space-time trellis coding," in *Proc. IEE Commun.*, Dec. 2005, pp. 1116–1124.

[84] S. X. Ng and Hanzo, "On the MIMO channel capacity of multi-dimensional signal sets," *IEEE Trans. Veh. Technol.*, vol. 55, no. 3, pp. 528–536, Mar. 2006.

[85] F. Brannstrom, L. K. Rasmussen, and A. J. Grant, "Convergence analysis and optimal scheduling for multiple concatenated codes," *IEEE Trans. Inform. Theory*, pp. 3354–3364, Sep. 2005.

[86] S. ten Brink, "Convergence of iterative decoding," *Inst. Elect. Eng. Electron. Lett.*, vol. 35, pp. 806–808, 1999.

[87] S. ten Brink, "Designing iterative decoding schemes with the extrinsic information transfer chart," *Int. J. Electron. Commun.*, vol. 54, no. 6, pp. 389–398, 2000.

[88] A. Grant, "Convergence of non-binary iterative decoding," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, San Antonio, TX, Nov. 2001, pp. 1058–1062.

[89] S. X. Ng, F. Guo, J. Wang, L.-L. Yang, and L. Hanzo, "Joint source-coding, channel coding and modulation schemes for AWGN and Rayleigh fading channels," *Inst. Elect. Eng. Electron. Lett.*, vol. 39, pp. 1259–1261, Aug. 21, 2003.

[90] J. Kliewer, S. X. Ng, and L. Hanzo, "Efficient computation of exit functions for non-binary iterative decoding," *IEEE Trans. Commun.*, vol. 54, no. 12, pp. 2133–2136, Dec. 2006.

[91] J. Hagenauer, "Source-controlled channel decoding," *IEEE Trans. Commun.*, vol. 43, no. 9, pp. 2449–2457, Sep. 1995.

[92] J. Wang, L.-L. Yang, and L. Hanzo, "Combined serially concatenated codes and MMSE equalisation: An EXIT chart aided perspective," in *Proc. 64th IEEE Vehicular Technology Conf. (VTC)*, Sep. 2006. [CD-ROM].

[93] J. Wang, S. X. Ng, A. Wolfgang, L.-L. Yang, S. Chen, and L. Hanzo, "Near-capacity three-stage MMSE turbo equalization using irregular convolutional codes," in *Proc. 4th Int. Symp. Turbo Codes (ISTC'06) in Connection With the 6th International ITG-Conf. Source and Channel Coding*, Munich, Germany, Apr. 2006.

ABOUT THE AUTHORS

Lajos Hanzo received the five-year Dipl.Ing. degree in electronics in 1976 and the doctorate degree in 1983. In 2004 he received the D.Sc. degree.

During his 30-year career in telecommunications he has held various research and academic posts in Hungary, Germany, and the U.K. Since 1986, he has been with the School of Electronics and Computer Science, University of Southampton, Southampton, U.K., where he holds the chair in telecommunications. He has coauthored 14 books on mobile radio communications and published about 700 research papers. Currently, he is directing an academic research team, working on a range of research projects in the field of wireless multimedia communications. He is an enthusiastic supporter of industrial and academic liaison and he offers a range of industrial courses.

Dr. Hanzo acted as TPC Chair of IEEE conferences, has presented keynote lectures, and has been awarded a number of distinctions. He is also an IEEE Distinguished Lecturer of both the Communications Society and the Vehicular Technology Society (VTS). Since 2005, he has been a Governor of the VTS. He is a Fellow of the Royal Academy of Engineering, U.K.

Jason P. Woodard was born in Northern Ireland in 1969. He received the B.A. degree in physics from Oxford University, Oxford, U.K., in 1991, and the M.Sc. degree (with distinction) in electronics and Ph.D. degree in speech coding from the University of Southampton, Southampton, U.K., in 1992 and 1995, respectively.

He then held a three-year postdoctoral fellowship, researching turbo coding techniques for the FIRST project within the European ACTS programme. In 1998, he joined the PA Consulting Group, Cambridge, U.K., and in 1999 he was a Founding Member of UbiNetics, a supplier of mobile communications test and IP solutions. Currently, he is working on the research and development of advanced wireless technologies for CSR, a leading supplier of single-chip wireless devices. He has published widely in wireless communications, including coauthoring two books.

Patrick Robertson was born in Edinburgh, Scotland, in 1966. He received the Dipl.-Ing. degree in electrical engineering from the Technical University of Munich, Munich, Germany, in 1989, and the Ph.D. degree from the University of the Federal Armed Forces, Munich, in 1995.

Since 1990, he has been working at the Institute for Communications Technology at the German Aerospace Centre (DLR), Oberpfaffenhofen, Germany. From 1990 to 1993, he shared this position with a part-time teaching post at the University of the Federal Armed Forces. He has been active in the fields of mobile packet transmission, digital terrestrial television (DVB-T), multimedia transmission, synchronization, turbo coding, and broadband wireless indoor communications networks. Since January 1999, he has been Leader of the research group "Broadband Systems and Navigation." He has published numerous scientific papers and holds numerous international patents in the areas of communications networks, mobile service discovery, indoor navigation, and systems for travel and tourist applications on wireless information devices. His current interests include navigation systems and algorithms and Bayesian inference techniques for context awareness within pervasive computing systems.