Adaptive Iterative Detection for Expediting the Convergence of a Serially Concatenated Unary Error Correction Decoder, Turbo Decoder and an Iterative Demodulator

Matthew F. Brejza, Wenbo Zhang, Robert G. Maunder, Bashir M. Al-Hashimi and Lajos Hanzo
School of ECS, University of Southampton, SO17 1BJ, UK
Email: {mfb2g09, wz4g11, rm, bmah, lh}@ecs.soton.ac.uk

***Abstract*—Unary Error Correction (UEC) codes constitute a recently proposed Joint Source and Channel Code (JSCC) family, conceived for alphabets having an infinite cardinality, whilst out-performing previously used Separate Source and Channel Codes (SSCCs). UEC based schemes rely on an iterative decoding process, which involves three decoding blocks when concatenated with a turbo code. Owing to this, following the activation of one of the three blocks, the next block to be activated must be chosen from the other two decoding block options. Furthermore, the UEC decoder offers a number of decoding options, allowing its complexity and error correction capability to be dynamically adjusted. It has been shown that iterative decoding convergence can be expedited by activating the specific decoding option that offers the highest Mutual Information (MI) improvement to computational complexity ratio. This paper introduces an iterative demodulator, which is shown to improve the associated error correction performance, while reducing the overall iterative decoding complexity. The challenge is that the iterative demodulator has to forward its soft-information to the other two iterative decoding blocks, and hence the corresponding MI improvements cannot be compared on a like-for-like basis. Additionally, we also propose a method of eliminating the logarithmic calculations from the adaptive iterative decoding algorithm, hence further reducing its implementational complexity without impacting its error correcting performance.**

I. Introduction

Wireless multimedia transmission has to be both bandwidth-efficient and resilient to transmission errors, motivating both source and channel coding. Traditionally, Separate Source and Channel Codes (SSCCs) have been employed for satisfying these requirements. For example, Elias Gamma (EG) codes [1] represent a typical source code, which may be concatenated with channel codes such as turbo codes [2]. However, SSCCs lead to capacity loss due to the residual redundancy that typically remains following source encoding, hence potentially limiting the error correction performance of the scheme [3]. In general, SSCC can be replaced by Joint Source and Channel Codes (JSCCs), which exploit the residual redundancy for the sake of achieving an improved error correction capability [4]. However, the source symbols produced by multimedia codecs such as the H.264 video encoder are approximately zeta distributed [3], with most symbols having a low value, but with infrequent symbols having high values of around 1000. The resultant high cardinality of the source alphabet prevents the employment of classic JSCCs, such as the Variable Length Error Correction (VLEC) [5], owing to the associated computational complexity. This potentially excessive complexity recently motivated the design of a JSCC known as the Unary Error Correction (UEC) code [3], which maintains a modest complexity, even when the cardinality of the source alphabet is infinite.

The authors wish to gratefully acknowledge the financial support of the ERSPC, Swindon UK under the auspices of grant EP/J015520/1 and EP/L010550/1, as well as the TSB, Swindon UK under the auspices of grant TS/L009390/1.

In [6], we concatenated the UEC encoder with a turbo encoder, corresponding to an iterative decoder comprising three decoding blocks. Following the activation of each decoding block, the next block to be activated must be chosen from the other two decoding options. Furthermore, the UEC decoder offers a number of decoding options [6], allowing its complexity versus error correction capability tradeoff to be dynamically adjusted. An adaptive iterative decoding algorithm was proposed in [6] for making a dynamic on-line selection of the decoding option to activate at each stage of the iterative decoding process. It was demonstrated that iterative decoding convergence can be expedited by activating the specific decoding option that offers the highest Mutual Information (MI) improvement to computational complexity ratio. This enables a frame to be decoded either at a reduced computational complexity or at a reduced Symbol Error Ratio (SER), when compared to a static activation order of the decoders.

In this paper, we extend our previous contribution of [6], by additionally concatenating the UEC code and turbo code with an iterative demodulator, producing an iterative receiver comprising four decoding blocks. We use EXtrinsic Information Transfer (EXIT) chart analysis for demonstrating that this iterative demodulator complements conveniently the turbo code, with the aid of reduced-complexity components, without increasing the E_b/N_0 at which iterative decoding convergence to a low SER becomes possible. We demonstrate that the introduction of iterative demodulation actually reduces the overall complexity of the iterative receiver. However, since the iterative demodulator is invoked for recovering different bits for forwarding to the UEC decoder and to the turbo component decoders, the extension of the adaptive iterative decoding algorithm of [6] to this scheme is not trivial. More specifically, the MI improvements associated with the iterative demodulator cannot be compared on a like-for-like basis with those offered by the three other decoding blocks. We propose a solution to this problem, which may also be applied in other multi-component iterative receivers. More specifically, we consider the operation of the iterative demodulator jointly with one of the turbo component decoders, allowing all decoding options or activation orders to be considered on the basis of MI improvements pertaining to the same bits. In this paper, we also explore the practical implementation of the adaptive algorithm advocated, proposing a simple approximation of the

MI measurement function. This reduces the associated implementational complexity by avoiding floating point logarithmic and exponential function evaluations, without significantly degrading the SER performance.

We commence in Section II by introducing the UEC code and our proposed schematic. Section III provides the EXIT chart [7] analysis of the decoding blocks, which form the basis of the adaptive algorithm. This section concludes by analyzing the implementational complexity of the proposed scheme, firstly quantifying its storage requirements, followed by detailing the proposed MI approximation. Error correction results are provided in Section IV, where the proposed scheme is found to offer up to 2 dB performance gain without increasing the required transmission energy, bandwidth, delay or decoder complexity. Section V offers our concluding remarks.

II. Transmitter and Receiver Scheme

In this section, we detail the operation of the UEC-Turbo-QPSK scheme of Fig. 1(a). The transmitter's operation is described in Section II-A, while the receiver is considered in Section II-B.

A. Transmitter

The transmitter design of the proposed joint source coding, channel coding and modulation scheme is shown in Fig. 1(a). This scheme is designed for conveying a vector $\mathbf{x} = [x_i]_{i=1}^{a}$ of a source symbols, each of which has a zeta-distributed integer value x_i in the of range one to infinity $x_i \in N_1$, and parameterized by $p_1 = P(x_i = 1)$ [3]. This distribution models the symbols generated by a H.264 video encoder [3], for example. These symbols are forwarded to the UEC source encoder, which comprises a unary encoder and a trellis encoder. The unary encoder produces a binary codeword $\mathbf{y}_i$ for each symbol x_i, where the length of the codeword is equal to the value of the symbol. The codewords are then concatenated for forming the bit vector $\mathbf{y} = [y_j]_{j=1}^{b}$, which is then encoded using the trellis encoder to obtain the bit vector $\mathbf{z} = [z_k]_{k=1}^{b}$. Here, $b = \sum_{i=1}^{a} x_i$ is the length of the bit vectors $\mathbf{y}$ and $\mathbf{z}$, which can vary from one frame to the next, owing to the varying lengths of the unary codewords produced by the UEC encoder. To elaborate fuurther, [6, Fig. 3] shows the $r = 6$-state UEC trellis, which is employed in the transmitter of Fig. 1(a). The UEC coding rate R_{o} is a function of the source distribution parameter p_1, where $p_1 = 0.797$ results in a coding rate of $R_{\text{o}} = 0.762$, for example.

Two replicas of the vector $\mathbf{z}$ are interleaved by the interleavers π_1 and π_2 to obtain the bit vectors $\mathbf{u}_{\text{u}}$ and $\mathbf{u}_{\text{l}}$, as shown in Fig. 1(a). Turbo channel encoding is performed upon $\mathbf{u}_{\text{u}}$ and $\mathbf{u}_{\text{l}}$ by a parallel concatenation of Unity Rate Code (URC) encoders [6] and the resultant bit vectors $\mathbf{v}_{\text{u}}$ and $\mathbf{v}_{\text{l}}$ are interleaved by a pair of interleavers, both having the same design π_3. The bits in the resultant vectors are paired up by a multiplexer to form the vector $\mathbf{w}$, which is QPSK modulated and transmitted over the channel using a natural mapping [8]. As we will show in Section IV, natural mapping facilitates improved error correction capability and/or reduced iterative decoding complexity, compared to Gray mapping. This results in an effective throughput given by $\eta = R_{\text{o}} R_{\text{i}} \log_2(M)$, where the use of the turbo code leads to $R_{\text{i}} = 0.5$ and we have $M = 4$ for QPSK modulation.

B. Receiver

The receiver of the proposed scheme is shown in Fig. 1(a). As seen in Fig. 1(a), the URC decoders convert the *a priori* Logarithmic Likelihood Ratio (LLR) vectors $\tilde{\mathbf{v}}_{\text{u}}^{\text{a}}$,$\tilde{\mathbf{u}}_{\text{u}}^{\text{a}}$, $\tilde{\mathbf{v}}_{\text{l}}^{\text{a}}$, $\tilde{\mathbf{u}}_{\text{l}}^{\text{a}}$ into the extrinsic LLR vectors $\tilde{\mathbf{v}}_{\text{u}}^{\text{e}}$,$\tilde{\mathbf{u}}_{\text{u}}^{\text{e}}$, $\tilde{\mathbf{v}}_{\text{l}}^{\text{e}}$, $\tilde{\mathbf{u}}_{\text{l}}^{\text{e}}$. Likewise the trellis decoder and the QPSK demodulator convert the LLR vectors $\tilde{\mathbf{z}}_{\text{o}}^{\text{a}}$, $\tilde{\mathbf{w}}_{\text{i}}^{\text{a}}$ into the extrinsic LLR vectors $\tilde{\mathbf{z}}_{\text{o}}^{\text{e}}$, $\tilde{\mathbf{w}}_{\text{i}}^{\text{e}}$, respectively. The UEC and URC decoders all rely on the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm for their operation.

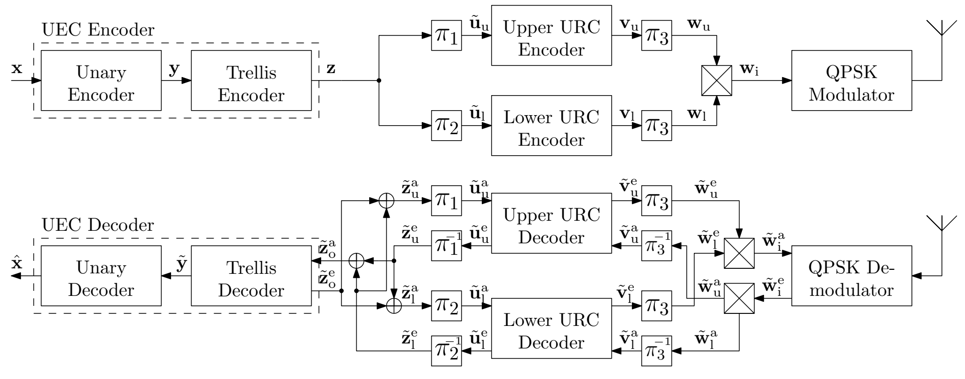
The QPSK demodulator [9] receives symbols from the channel. It bi-directionally exchanges LLRs with the URC decoders, where the extrinsic LLR vector $\tilde{\mathbf{w}}_{\text{i}}^{\text{e}}$ is demultiplexed and deinterleaved through π_3^{-1}, before being provided as *a priori* LLR vectors $\tilde{\mathbf{v}}_{\text{u}}^{\text{a}}$ and $\tilde{\mathbf{v}}_{\text{l}}^{\text{a}}$ to the upper and lower URC decoders, respectively. The iterative demodulator also relies on natural mapping [8], where the upper and lower URC decoders exchange their extrinsic LLR vectors $\tilde{\mathbf{v}}_{\text{u}}^{\text{e}}$ and $\tilde{\mathbf{v}}_{\text{l}}^{\text{e}}$ with the demodulator through π_3 and a multiplexer, providing the demodulator with the *a priori* LLR vector $\tilde{\mathbf{w}}_{\text{i}}^{\text{a}}$.

Fig. 1(a) shows that the *a priori* LLR vectors $\tilde{\mathbf{z}}_{\text{o}}^{\text{a}}$, $\tilde{\mathbf{z}}_{\text{u}}^{\text{a}}$ and $\tilde{\mathbf{z}}_{\text{l}}^{\text{a}}$ provided for each of the three decoders are obtained as the sum of the extrinsic LLR vectors most recently generated by the other two decoders, namely we have $\tilde{\mathbf{z}}_{\text{o}}^{\text{a}} = \tilde{\mathbf{z}}_{\text{u}}^{\text{e}} + \tilde{\mathbf{z}}_{\text{l}}^{\text{e}}$, $\tilde{\mathbf{z}}_{\text{u}}^{\text{a}} = \tilde{\mathbf{z}}_{\text{o}}^{\text{e}} + \tilde{\mathbf{z}}_{\text{l}}^{\text{e}}$ and $\tilde{\mathbf{z}}_{\text{l}}^{\text{a}} = \tilde{\mathbf{z}}_{\text{o}}^{\text{e}} + \tilde{\mathbf{z}}_{\text{u}}^{\text{e}}$. These LLR vectors comprise b number of LLRs, which pertain to the corresponding bits of $\mathbf{z}$. At the start of the iterative decoding process, the extrinsic LLR vectors $\tilde{\mathbf{z}}_{\text{o}}^{\text{e}}$, $\tilde{\mathbf{z}}_{\text{u}}^{\text{e}}$ and $\tilde{\mathbf{z}}_{\text{l}}^{\text{e}}$ are initialized with zero-valued LLRs. The interleavers π_1, π_2 and the deinterleavers π_1^{-1}, π_2^{-1} match with the interleavers in the transmitter, and are used for translating the LLR vectors $\tilde{\mathbf{z}}_{\text{u}}^{\text{a}}$, $\tilde{\mathbf{z}}_{\text{l}}^{\text{a}}$, $\tilde{\mathbf{u}}_{\text{u}}^{\text{e}}$, $\tilde{\mathbf{u}}_{\text{l}}^{\text{e}}$ into the vectors $\tilde{\mathbf{u}}_{\text{u}}^{\text{a}}$, $\tilde{\mathbf{u}}_{\text{l}}^{\text{a}}$, $\tilde{\mathbf{z}}_{\text{u}}^{\text{e}}$, $\tilde{\mathbf{z}}_{\text{l}}^{\text{e}}$, respectively.

As shown in [6], the number of states employed by the UEC trellis decoder can be selected independently of the number of states employed in the UEC trellis encoder. Increasing the number of states r employed by the UEC trellis decoder in this way has the benefit of improving its error correction capability, albeit this is achieved at the cost of increasing its complexity [3]. In Section III-B, we will exploit this for dynamically adjusting r for the sake of striking an attractive trade-off between the UEC trellis decoder's complexity and its error correction capability.

During the iterative decoding process, the iterative operation of the URC1, URC2, UEC decoders and of the demodulator seen in Fig. 1(a) may be performed using a wide variety of different decoder activation orderings. For the sake of conceptional simplicity, a fixed decoder activation order may be employed, in which the URC1, URC2, UEC decoders and the demodulator of Fig. 1(a) are activated using a regular activation order repeated periodically, namely [demod, URC1, URC2, UEC, demod, URC1,...]. Alternatively, we may employ a dynamic decoder activation order, in which an on-line decision is made at each stage of the iterative decoding process, in order to adaptively and dynamically select which of the URC1, URC2, UEC decoders or whether the demodulator of Fig. 1(a) should be activated next. This is explored in Section III using a novel 3D EXIT chart aided technique, in order to expedite iterative decoding convergence towards the Maximum Likelihood (ML) decoder's performance.

Following the achievement of iterative decoding convergence, the UEC trellis decoder of Fig. 1(a) generates the vector of *a posteriori* LLRs $\tilde{\mathbf{y}}^{\text{p}}$, which pertain to the corresponding unary-encoded bits in the vector $\mathbf{y}$. Finally, $\tilde{\mathbf{y}}^{\text{p}}$ can be unary decoded in order to obtain the symbol vector $\hat{\mathbf{x}}$, as detailed in [3].



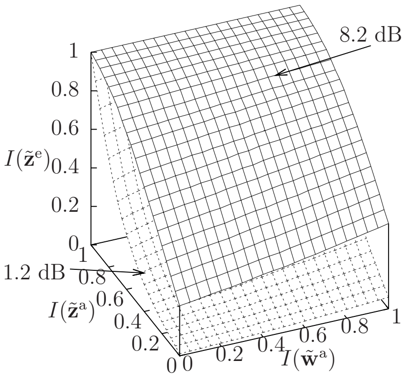


Fig. 1. (a) Schematic of the proposed transmitter (top), as well as the proposed receiver (bottom). (b) The EXIT function $I(\tilde{\mathbf{z}}^{\mathrm{e}}_{\mathrm{u}}) = f_{\mathrm{D-URC}}[I(\tilde{\mathbf{z}}^{\mathrm{a}}), I(\tilde{\mathbf{w}}^{\mathrm{a}}), E_{\mathrm{b}}/N_0]$ for E_{b}/N_0 = 1.2 dB and E_{b}/N_0 = 8.2 dB.

III. Adaptive Decoder Activation Order Algorithm

This section details the proposed extension to the adaptive iterative decoding algorithm of [6] for the sake of facilitating its contribution with schemes relying on additional serially concatenated blocks, such as the proposed concatenation of an iterative source and channel code with an iterative demodulator. This algorithm is best suited for scenarios which are complexity-limited, since it allows the receiver to adapt the decoder activation order to make best use of the limited resources. We begin in Section III-A by considering the EXIT functions of each decoder, which are used by the proposed algorithm for quantifying the benefit of activating the corresponding decoding block at a particular point in the iterative decoding process. Following this, Section III-B details the decision process used by the proposed adaptive algorithm. Section III-C quantifies the storage requirements of the algorithm, while Section III-D conceives a technique for eliminating the algorithm's complex logarithmic and exponential operations without a reducing its performance.

A. EXIT Function Analysis

EXIT functions [7] rely on the MI to quantify the quality of the extrinsic information output by a decoding block, as a function of the quality of its input *a priori* information, as well as the the channel Signal-to-Noise Ratio (SNR) if the decoding block performs demodulation, equalization or multi-user detection, for example.

The following sections describe the EXIT functions for the different components of the scheme shown in Fig. 1(a), with the UEC EXIT function described in Section III-A1, the URC described in Section III-A2, and the combined demodulator and URC EXIT function described in Section III-A3. In common with our previous work [6], all of the EXIT functions are defined in terms of $I(\tilde{\mathbf{z}})$ and $I(\tilde{\mathbf{w}})$, rather than $I(\tilde{\mathbf{u}})$ and $I(\tilde{\mathbf{v}})$.

1) UEC Decoder: The operation of the UEC decoder is only dependent on a single input vector of *a priori* LLRs $\tilde{\mathbf{z}}^{\mathrm{a}}_{\mathrm{o}}$, which is used for producing both the output extrinsic LLR vector $\tilde{\mathbf{z}}^{\mathrm{e}}_{\mathrm{o}}$ and the *a posteriori* LLR vector $\tilde{\mathbf{y}}$. The UEC EXIT functions may be expressed as $I(\tilde{\mathbf{z}}^{\mathrm{e}}_{\mathrm{o}}) = f_{\mathrm{UEC}}[I(\tilde{\mathbf{z}}^{\mathrm{a}}_{\mathrm{o}}), r]$, where $I(\tilde{\mathbf{z}})$ is the MI of the LLR vector $\tilde{\mathbf{z}}$, and r is the number of states in the UEC trellis decoder. The operation of the UEC decoder may be described by a series of 2D EXIT functions, with a separate EXIT function for each considered value of r. The family of 2D EXIT functions for the UEC decoder are exemplified in [10, Fig. 3] for $p_1 \in \{0.7, 0.8, 0.9\}$ and $r \in \{4, 6, 8, 10, 32\}$.

2) URC Decoder: The URC decoder is dependent on the *a priori* LLR vector inputs $\tilde{\mathbf{w}}^{\mathrm{a}}$ and $\tilde{\mathbf{z}}^{\mathrm{a}}$, which it used for providing two extrinsic output LLR vectors $\tilde{\mathbf{w}}^{\mathrm{e}}$ and $\tilde{\mathbf{z}}^{\mathrm{e}}$, where the $_{\mathrm{u}}$ and $_{\mathrm{l}}$ subscripts refer specifically to the upper and lower URC decoders, respectively. Note however that it is only necessary for the MI of the $\tilde{\mathbf{z}}^{\mathrm{e}}$ output to be considered by the adaptive algorithm, as it will be described in Section III-B. The EXIT function for the $\tilde{\mathbf{z}}^{\mathrm{e}}$ output of a URC decoder can therefore be expressed as a function depending on two input MI values, leading to a 3D EXIT function. More specifically, we have $I(\tilde{\mathbf{z}}^{\mathrm{e}}) = f_{\mathrm{URC}}[I(\tilde{\mathbf{z}}^{\mathrm{a}}), I(\tilde{\mathbf{w}}^{\mathrm{a}})]$, where $I(\tilde{\mathbf{z}}^{\mathrm{a}})$ and $I(\tilde{\mathbf{w}}^{\mathrm{a}})$ are the mutual information of $\tilde{\mathbf{z}}^{\mathrm{a}}$ and $\tilde{\mathbf{w}}^{\mathrm{a}}$, respectively.

3) Demodulator and URC Decoder: The iterative QPSK demodulator and one of the URC decoders can be considered as a single block for the purposes of the proposed algorithm. This is because the operation of the demodulator will always be followed by activating one of the URC decoders, so that the resultant information $\tilde{\mathbf{w}}^{\mathrm{e}}_{\mathrm{i}}$ gleaned from the demodulator can be propagated to the rest of the iterative decoder. This approach takes the demodulator into account, but allows an EXIT function to be produced that characterizes the expected MI improvement of the signal $\mathbf{z}^{\mathrm{e}}$. Since both the URC and UEC EXIT functions also characterize the improvement of $\mathbf{z}^{\mathrm{e}}$, this allows like-for-like comparisons to be made between the three different types of decoding blocks.

The joint demodulator and URC decoder EXIT function depends on the channel's E_{b}/N_0, as well as on the *a priori* LLR vector $\tilde{\mathbf{z}}^{\mathrm{a}}_{\mathrm{u}}$ or $\tilde{\mathbf{z}}^{\mathrm{a}}_{\mathrm{l}}$ provided for the URC decoder and on the *a priori* LLR vector $\tilde{\mathbf{w}}^{\mathrm{a}}_{\mathrm{i}}$ fed back to the demodulator. This can be expressed as $I(\tilde{\mathbf{z}}^{\mathrm{e}}) = f_{\mathrm{D-URC}}[I(\tilde{\mathbf{z}}^{\mathrm{a}}), I(\tilde{\mathbf{w}}^{\mathrm{a}}_{\mathrm{i}}), E_{\mathrm{b}}/N_0]$, which allows a series of 3D EXIT functions to be produced, with a specific 3D EXIT function for each E_{b}/N_0 value considered. The EXIT functions recorded for two E_{b}/N_0 values are shown in Fig. 1(b) for the combination of natural mapping aided QPSK and a URC decoder having 8 states.

B. Adaptive activation algorithm

Upon starting to detect a new frame, the receiver first has to activate the iterative demodulator, followed by one of the URC blocks in order to disseminate the information received from the channel. Following this, it may decide from several options as to which decoding block to operate next. In general, the decoding activation options that can be selected based on Fig 1(a) are the upper URC decoder (URC1), the lower URC decoder (URC2), the demodulator followed by the upper URC decoder (URC1+demod), the demodulator followed by the lower URC decoder (URC2+demod), or the UEC decoder (UEC), where selecting the UEC presents the option of operating with different numbers of states r.

During the iterative decoding process, the benefit of each decoding option is quantified using the expected improvement to the quality of the extrinsic LLR vector that it provides. This is achieved by first measuring the MI of the most-recently generated values of the *a priori* LLR vectors $\tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{a}}$, $\tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{a}}$, $\tilde{\mathbf{z}}_{\mathrm{l}}^{\mathrm{a}}$, $\tilde{\mathbf{w}}_{\mathrm{u}}^{\mathrm{a}}$, and $\tilde{\mathbf{w}}_{\mathrm{l}}^{\mathrm{a}}$, using the technique described in Section III-D. These MIs may be used as inputs to the EXIT functions of Section III-A, which may be generated off-line and stored in Look-Up Tables (LUTs), as will be described in Section III-C. These predict the MI of the extrinsic LLR vector that would be generated by each decoding option This can be compared to the MI of the currently available extrinsic LLR vector that would be replaced to make an estimation of the potential MI improvement offered by selecting this decoding option. Note that additions are used for obtaining the LLR vectors $\tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{a}} = \tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{e}} + \tilde{\mathbf{z}}_{\mathrm{l}}^{\mathrm{e}}$, $\tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{a}} = \tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{e}} + \tilde{\mathbf{z}}_{\mathrm{l}}^{\mathrm{e}}$ and $\tilde{\mathbf{z}}_{\mathrm{l}}^{\mathrm{a}} = \tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{e}} + \tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{e}}$. The MI of the *a priori* LLR vectors $\tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{a}}$, $\tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{a}}$ and $\tilde{\mathbf{z}}_{\mathrm{l}}^{\mathrm{a}}$ can be estimated without tentatively adding the extrinsic LLR vectors $\tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{e}}$, $\tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{e}}$ and $\tilde{\mathbf{z}}_{\mathrm{l}}^{\mathrm{e}}$. Instead, we can use the J-function proposed in [11], which takes as its input two MI values, and outputs the joint MI of the two inputs. This allows us to use $I(\tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{a}}) = J[I(\tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{e}}), I(\tilde{\mathbf{z}}_{\mathrm{l}}^{\mathrm{e}})]$ as an input to $I(\tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{e}}) = f_{\mathrm{UEC}}[I(\tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{a}}), r]$ in order to quantify the potential benefit of activating the UEC decoder. Likewise, $I(\tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{a}}) = J[I(\tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{e}}), I(\tilde{\mathbf{z}}_{\mathrm{l}}^{\mathrm{e}})]$ and $I(\tilde{\mathbf{w}}_{\mathrm{u}}^{\mathrm{a}})$ can be used as inputs to $I(\tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{e}}) = f_{\mathrm{URC}}[I(\tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{a}}), I(\tilde{\mathbf{w}}_{\mathrm{u}}^{\mathrm{a}})]$, for predicting the potential benefit of operating the URC1 decoder. Similarly, $I(\tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{a}}) = J[I(\tilde{\mathbf{z}}_{\mathrm{l}}^{\mathrm{e}}), I(\tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{e}})]$, $I(\tilde{\mathbf{w}}_{\mathrm{i}}^{\mathrm{e}})$ and the channel SNR can be used as inputs to $I(\tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{e}}) = f_{\mathrm{D-URC}}[I(\tilde{\mathbf{z}}_{\mathrm{u}}^{\mathrm{a}}), I(\tilde{\mathbf{w}}_{\mathrm{i}}^{\mathrm{e}}), E_{\mathrm{b}}/N_0]$, for predicting the expected benefit of operating the demodulator followed by URC1. By subtracting the MI of the currently avaliable extrinsic LLR vector from the expected MI of the vector that would replace it, we obtain the expected MI improvement for each decoding option. By selecting the next decoding option with consideration of its expected MI improvement, it allows us to give priority to those blocks which have not been activated recently.

The algorithm for selecting which of the available decoding options to perform next (UEC, URC1, URC2, URC1+demod, URC2+demod) also quantifies of the cost of each the options. More specifically, we consider using the number of Add-Compare-Select (ACS) [10] operations needed per bit to perform each option. The number of ACS operations for the URC and UEC decoders are approximately proportional to the number of states in the trellis. A 4-state and an 8-state URC require 112 and 223 ACS operations per bit of $\mathbf{z}$, respectively. Meanwhile an $r = \{2, 4, 6, 8, 10\}$ state UEC requires $\{50, 108, 166, 224, 282\}$ ACS operations per bit of $\mathbf{z}$, respectively. The demodulator has a much lower complexity, requiring only 11 ACS operations per bit. This results in 123 and 234 ACS operations per bit of $\mathbf{z}$, when the demodulator is combined with a 4-state and 8-state URC, respectively. A full breakdown of the operations required by the decoding blocks is available in [6, Table III]. Upon dividing the expected MI improvement by the number of ACS operations needed per bit for each of the decoding options, we arrive at a score for each decoding option. The algorithm then picks the specific option with the highest score as the block which should be operated next.

C. Storage Requirements

In this section, we consider the memory required for the storage of the EXIT function LUTs, which are required for the adaptive iterative decoding algorithm of Section III-B to function. As described in Section III-B, three sets of EXIT function LUTs are required: one set for the UEC code $I(\tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{e}}) = f_{\mathrm{UEC}}[I(\tilde{\mathbf{z}}_{\mathrm{o}}^{\mathrm{a}}), r]$, with one 2D EXIT function for each of the $G_r = 5$ number of states $r \in \{2, 4, 6, 8, 10\}$ considered; one 3D EXIT function LUT for the URC codes $I(\tilde{\mathbf{z}}^{\mathrm{e}}) = f_{\mathrm{URC}}[I(\tilde{\mathbf{z}}^{\mathrm{a}}), I(\tilde{\mathbf{w}}^{\mathrm{a}})]$; and one set of 3D EXIT functions $I(\tilde{\mathbf{z}}^{\mathrm{e}}) = f_{\mathrm{D-URC}}[I(\tilde{\mathbf{z}}^{\mathrm{a}}), I(\tilde{\mathbf{w}}^{\mathrm{a}}), E_{\mathrm{b}}/N_0]$ for the combined demodulator and URC decoders, where there is one 3D EXIT function for each of the 24 E_{b}/N_0 values considered. These G_{S}=24 different E_{b}/N_0 values were selected in the range 1.2 dB to 8.2 dB, where EXIT functions for lower E_{b}/N_0 values are not stored, since successful decoding is not possible in this case, while EXIT functions for higher E_{b}/N_0 values are not needed, since the decoding complexity is low in this case, hence reducing the benefit of the adaptive algorithm.

For all the EXIT functions, G_{I}=21 different *a priori* MI values $\{0, 0.05, 0.1, ..., 1\}$ were selected for each input. The results of Section IV-B compare the iterative decoding performance, when quantization levels of $B \in \{4, 6, 8\}$ bits are used for representing each value in the EXIT function LUT, resulting in overall memory requirements of 5.5 kB, 8.25 kB and 11 kB, respectively. This storage requirement S can be generalized to $S[\text{bits}] = B(G_{\mathrm{I}}^2(G_{\mathrm{S}} + 1) + G_{\mathrm{I}}G_r)$. As a comparison, the combined memory requirement for three different 6144 bit random interleavers is 30 kB, demonstrating that the memory requirement of the proposed algorithm is not excessive, when compared to the original memory requirement of the scheme in Fig. 1(a).

D. MI Measurement

The adaptive iterative decoding algorithm of Section III-B relies on measuring the MI of the LLR vectors provided by the various blocks of the decoder. Using the averaging method [7], the MI of an LLR vector may be is estimated without any knowledge of the true bit values. The MI is calculated as a function of the entropies that are implied by the b LLRs in the vector [7], according to $I(\tilde{\mathbf{z}}) = \frac{1}{b}\sum_{k=1}^{b} 1 - H_{\mathrm{b}}(\tilde{z}_k)$, where $H_{\mathrm{b}}(\tilde{z}_k)$ is the binary entropy function.

The binary entropy function $H_{\mathrm{b}}(\tilde{z}_k)$ is comprised of several log and exp functions, which imposes a high implementation complexity. As a result, it is desirable to approximate this function. Fig. 2 shows that the function $1 - H_{\mathrm{b}}(\tilde{z}_k)$ resembles an inverted Gaussian distribution. Motivated by this, we propose to approximate it using a linear approximation. As shown in Fig. 2, this approximation is characterized by a straight line, which is mirrored for positive and negative LLR values, and that is clipped such that it has a maximum MI value of 1, and the minimum MI value of 0. This results in a linear piece-wise approximation of 5 segments, and can be written as $I(\tilde{z}_k) = \max(\min(m|\tilde{z}_k| + c, 1), 0)$, where m and c represent the gradient and intercept point of the line. To choose the optimum values for m and c, LLRs were generated having a range of target MI values in the range of 0 to 1. Following this, the approximate MI measurement function was fitted to these LLRs, producing an estimate for the MI of each group of LLRs. The slope m and intercept c values were chosen for the approximation as those, which produced the minimum aggregate error between the approximate MI values and the exact MI values. In this case, $m = 0.24$ and $c = -0.06$ were found to be optimum. To reduce the number of calculations needed, instead of scaling and shifting the MI of each LLR before averaging, we can transform the operation, so that the clipped LLRs are averaged, before this average is scaled by m and shifted by c to get the final MI of the vector. These techniques result in negligible complexity for the

MI measurement compared to the operation of the decoding blocks.

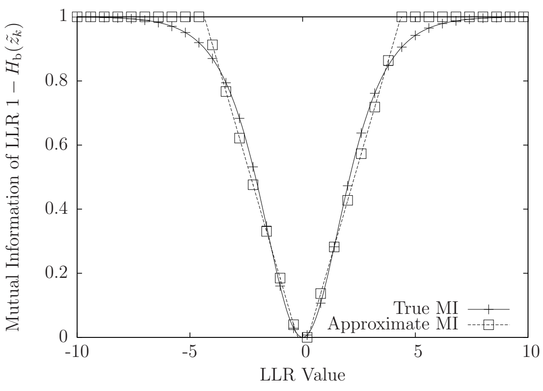


Fig. 2. The relationship between the true MI and the approximate MI

IV. Results

In this section, the error correcting performance of the proposed scheme using the adaptive iterative decoding algorithm is compared to that of various benchmarkers. This section will first describe the nature of these benchmarkers in Section IV-A, before presenting the results in Section IV-B.

A. Considered Schemes

In this section we contrast the proposed scheme of Section II to a number of benchmarkers. A summary of the benchmarkers is shown in Table I and more detailed information is available in [6]. For each scheme, we will consider both a 4-state and an 8-state URC design for the upper and lower URC decoder producing turbo decoders, which are referred to as Turbo4 and Turbo8 from here on. Each of the schemes can also be operated with or without the iterative demodulator, where activation of the iterative demodulator is denoted with the -QPSK suffix. In the case when the QPSK demodulator is not operated iteratively, Gray mapping is employed since it provides superior performance to natural mapping in this case [8]. Table I shows the E_b/N_0 capacity bound, which is the theoretical limit for reliable communication of the given effective throughput η, the E_b/N_0 area bound which is the limit at which the area beneath the EXIT functions match each other, and finally the E_b/N_0 tunnel bound, which is the threshold where the EXIT charts exhibit an open tunnel. These bounds indicate successively stricter limits for the E_b/N_0, where a low SER becomes possible for each scheme. Since all of the schemes have the same effective throughput of $\eta = 0.762$, our comparisons between them are fair in terms of transmission energy, delay and bandwidth.

1) UEC-Turbo(-QPSK): This is the proposed scheme of Fig. 1(a), which may be operated with or without the aid of the adaptive iterative decoding algorithm advocated. The tunnel bound of Table I suggests that the UEC-Turbo8 scheme of our previous work [6] would marginally out-perform UEC-Turbo4-QPSK by 0.1 dB, but only in the absence of an iterative decoding complexity limit. Although the iterative demodulator of the proposed scheme imposes only a modest complexity, it actually facilitates a significant overall complexity reduction since it allows the less complex 4-state URC to be used, suggesting that it will out-perform the UEC-Turbo8 scheme in the presence of an iterative decoding complexity limit.

2) EG-URC-Turbo(-QPSK): This scheme is a classic SSCC benchmarker, where the unary encoder of Fig. 1(a) is replaced with an EG code for source coding. Furthermore, the UEC trellis code of Fig. 1(a) is replaced with an accumulator, which is an $r = 2$ state URC encoder. This is required for converting the vector of non-equiprobable bits $\mathbf{y}$ into a vector of equiprobable bits $\mathbf{z}$ [6]. During decoding, each of the three URC decoders are operated in turn. In this scheme the outer URC decoder is provided with *a priori* information pertaining to the bit value probabilities in $\mathbf{y}$, which allows some of the residual redundancy to be exploited for error correction.

3) EG-Turbo(-QPSK): This SSCC benchmarker replaces the UEC code of Fig. 1(a) with an EG code. This results in a variation of the EG-URC-Turbo scheme, which omits the URC code. This scheme suffers from a significant capacity loss due to the non-equiprobable bits in $\mathbf{z}$ [6]. However, this scheme has a low complexity due to the reduced number of iterative blocks, which are operated in the decoder.

4) Unary-Turbo(-QPSK): This is another SSCC benchmarker, which has non-equiprobable bits at $\mathbf{y}$, and therefore also suffers a from capacity loss. Compared to the proposed scheme of Fig. 1(a), the UEC code has been replaced by a unary code. It is therefore similar to the EG-Turbo scheme, but uses a unary encoder instead of a EG encoder.

B. SER Performance

In this section, we consider a limited complexity-receiver, where there is a limited amount of computational resource. Fig. 3 provides SER plots for each of the schemes chosen and for a range of complexity limits, where complexity is quantified in terms of ACS operations per bit of $\mathbf{z}$. Results are provided for the transmission of frames comprising $a = 650$ symbols over an uncorrelated narrowband Rayleigh fading channel. The complexity limits imposed are 1000, 2000 and 3000 operations per bit of $\mathbf{z}$, which allow the relative performance of the various schemes to be compared for a range of available decoder resources. If the iterative scheme of Fig. 1(a) operates each of the blocks successively in turn, these complexity limits allow approximately 3, 6 and 9 iterations to be completed, when 4 states are used for the UEC and URC decoders.

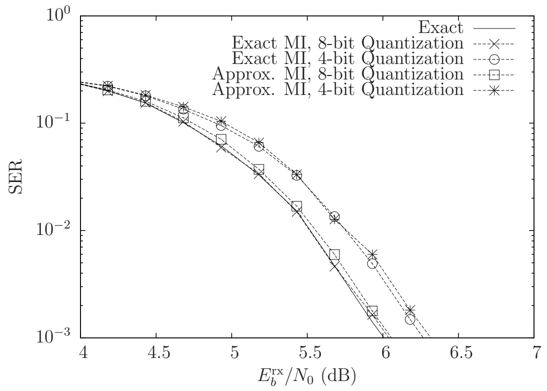
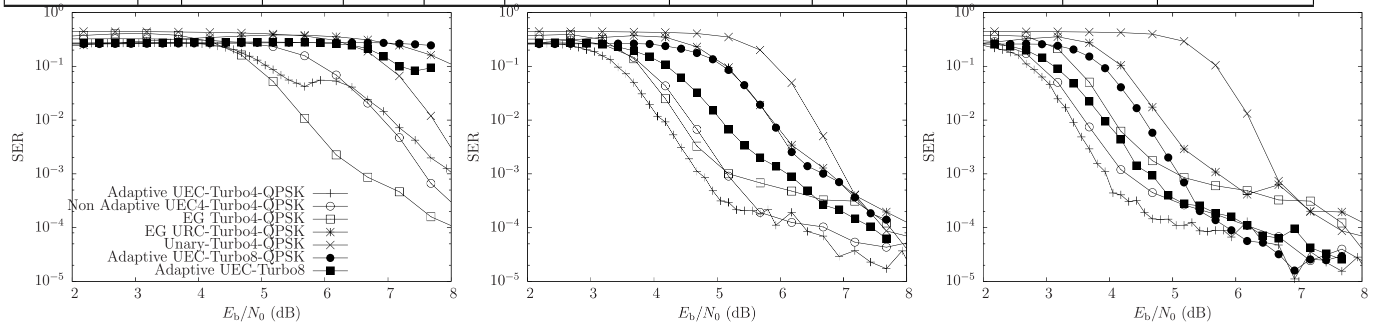


Fig. 4. SER performance of the adaptive algorithm for different approximations of UEC-Turbo8-QPSK and $C = 2000$ ACS ops/bit. Frame length = 650 symbols

1) Effect of the Iterative Demodulator: As shown in Fig. 3, when the scheme of Fig. 1(a) uses 4-state URCs and the iterative demodulator (UEC-Turbo4-QPSK), it performs better than the UEC-Turbo8 scheme of [6], which uses 8-state URCs without iterative demodulation. Although the EXIT chart analysis of Section IV-A suggests that the ultimate performance of the UEC-Turbo8 scheme would be 0.1 dB better, the UEC-Turbo4-QPSK arrangement is seen to be 2 dB superior, when a maximum complexity of $C = 2000$ ops/bit of $\mathbf{z}$ is imposed.

TABLE I
COMPARISON BETWEEN THE CHARACTERISTICS OF THE DIFFERENT BENCHMARKERS

Scheme	r	R_o	A_o	R_i	η	$E_b/N_0[dB]$ capacity bound	$E_b/N_0[dB]$ area bound	$E_b/N_0[dB]$ tunnel bound			
								Turbo4	Turbo4-QPSK	Turbo8	Turbo8-QPSK
UEC-Turbo	2	0.762	0.934	0.500	0.762	0.84	2.48	3.1	2.8	2.7	3.1
	4		0.808				1.27	2.3	1.6	1.8	2.0
	6		0.783				1.04	1.8	1.3	1.3	1.8
	8		0.774				0.95	1.8	1.3	1.2	1.7



(a) Complexity = 1000 ACS ops/bit (b) Complexity = 2000 ACS ops/bit (c) Complexity = 3000 ACS ops/bit

Fig. 3. SER performance for the adaptive scheme and several benchmarkers under different complexity limits. Frame length = 650 symbols

This may be attributed to the fact that the 4-state URC decoder imposes about half the complexity of the 8-state URC decoder, while the added complexity of the iterative demodulator is minimal in comparison.

The proposed UEC-Turbo4-QPSK scheme performs particularly well under low complexity limits, since its advantage of reduced complexity is diminished, as the complexity limit is increased. Owing to this, the gap between the UEC-Turbo8 and UEC-Turbo4-QPSK schemes reduces to 1.2 dB for a complexity limit of $C = 3000$ ACS ops/bit of $\mathbf{z}$.

2) Low Complexity Schemes vs. High Performance Schemes: Fig. 3 also compares the proposed adaptive UEC-Turbo4-QPSK scheme to other benchmarkers. Since the proposed adaptive algorithm allows for a reduced decoding complexity, it has a performance advantage over the set of high-performance benchmarkers. However, under the low complexity limit of $C = 1000$ ACS per bit of $\mathbf{z}$, the less complex benchmarkers, such as the EG-Turbo scheme, exhibit a 1.5 dB gain over the more complex near-capacity schemes. However, when the complexity limit is increased to $C = 2000$ ACS per bit of $\mathbf{z}$, these low complexity schemes suffer a performance loss, while the proposed adaptive UEC-Turbo4-QPSK scheme offers a modest, but non-negligible performance gain of 0.25 dB, without any increase in transmission energy, delay, bandwidth or decoder complexity.

3) Effects of Approximation: Fig. 4 characterizes the performance erosion imposed upon the proposed adaptive UEC-Turbo4-QPSK scheme, when EXIT chart LUT quantization is used, as well as when approximate MI measurements are used. It can be seen in Fig. 4 that 6-bit and 8-bit-quantization imposes only a negligible degradation on the SER performance, while 4-bit-quantization reduces the gain of the proposed scheme by 0.2 dB. The approximation invoked for measuring the MI also has a negligible impact on the proposed scheme, hence this is recommended for hardware implementation.

V. CONCLUSIONS

In this paper we have demonstrated that the adaptive iterative decoding algorithm of [6] can be further developed for reducing the complexity of an iterative decoder with a four stage concatenation, including an iterative demodulator. This iterative demodulator also allows the employment of a URC code with fewer states, resulting in an overall complexity, whilst beneficially increasing the performance of the scheme in complexity-limited scenarios. The proposed adaptive algorithm allows low SERs to be achieved over a range of complexity limits, while the benchmarkers either favor low complexity limits or high complexity limits exclusively. This paper also proposes an approximation to the MI measurement used in the adaptive algorithm, facilitating its hardware implementation with a low overhead.

REFERENCES

[1] P. Elias, "Universal Codeword Sets and Representations of the Integers," *IEEE Transactions on Information Theory*, vol. 21, no. 2, pp. 194–203, Mar. 1975.

[2] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error Correcting Coding and Decoding: Turbo Codes," in *Proceedings of the IEEE International Conference on Communications*, vol. 2, Geneva, Switzerland, 1993, pp. 1064–1070.

[3] R. G. Maunder, W. Zhang, T. Wang, and L. Hanzo, "A Unary Error Correction Code for the Near-Capacity Joint Source and Channel Coding of Symbol Values from an Infinite Set," *IEEE Transactions on Communications*, vol. 61, pp. 1977–1987, 2013.

[4] J. Hagenauer and N. Gortz, "The Turbo Principle in Joint Source-Channel Coding," in *Proceedings 2003 IEEE Information Theory Workshop*. IEEE, 2003, pp. 275–278.

[5] L. Hanzo, R. G. Maunder, J. Wang, and L.-L. Yang, *Near-Capacity Variable-Length Coding*. Chichester, UK: John Wiley & Sons, Ltd, Oct. 2010.

[6] W. Zhang, Y. Jia, X. Meng, M. Brejza, R. G. Maunder, and L. Hanzo, "Adaptive Iterative Decoding for Expediting the Convergence of Unary Error Correction Codes," *IEEE Transactions on Vehicular Technology*, pp. 1–1, May 2014.

[7] J. Hagenauer, "The EXIT Chart-Introduction to Extrinsic Information Transfer in Iterative Processing," *Proc. 12th European Signal Processing Conference*, 2004.

[8] N. F. Kiyani and J. H. Weber, "Iterative Demodulation and Decoding for Rotated MPSK Constellations with Convolutional Coding and Signal Space Diversity," in *2007 IEEE 66th Vehicular Technology Conference*. IEEE, Sep. 2007, pp. 1712–1716.

[9] M. Valenti and S. Cheng, "Iterative Demodulation and Decoding of Turbo-Coded M-ary Noncoherent Orthogonal Modulation," *Selected Areas in Communications, IEEE Journal on*, 2005.

[10] W. Zhang, R. G. Maunder, and L. Hanzo, "On the Complexity of Unary Error Correction Codes for the Near-Capacity Transmission of Symbol Values from an Infinite Set," in *IEEE Wireless Communications and Networking Conference 2013*, no. IID, Oct. 2012, pp. 1–6.

[11] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of Low-Density Parity-Check Codes for Modulation and Detection," *IEEE Transactions on Communications*, vol. 52, no. 4, pp. 670–678, Apr. 2004.