Myburgh and Olivier *EURASIP Journal on Advances in Signal Processing* 2013, **2013**:79
http://asp.eurasipjournals.com/content/2013/1/79

EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH Open Access

A primer on equalization, decoding and non-iterative joint equalization and decoding

Hermanus C Myburgh[1*] and Jan C Olivier[2]

Abstract

In this article, a general model for non-iterative joint equalization and decoding is systematically derived for use in systems transmitting convolutionally encoded BPSK-modulated information through a multipath channel, with and without interleaving. Optimal equalization and decoding are discussed first, by presenting the maximum likelihood sequence estimation and maximum *a posteriori* probability algorithms and relating them to equalization in single-carrier channels with memory, and to the decoding of convolutional codes. The non-iterative joint equalizer/decoder (NI-JED) is then derived for the case where no interleaver is used, as well as for the case when block interleavers of varying depths are used, and complexity analyses are performed in each case. Simulation results are performed to compare the performance of the NI-JED to that of a conventional turbo equalizer (CTE), and it is shown that the NI-JED outperforms the CTE, although at much higher computational cost. This article serves to explain the state-of-the-art to students and professionals in the field of wireless communication systems, presenting these fundamental topics clearly and concisely.

Keywords: Equalizer, Decoder, Joint equalization/decoding, Computational complexity.

1 Introduction

Equalization and decoding are two essential aspects of any wireless communication system. The equalizer is tasked with reversing the effect of the communication channel on the transmitted information signal, while the decoder receives the equalized symbol sequence and attempts to correct errors that might have been caused during transmission.

The Bahl–Cocke–Jelinek–Raviv (BCJR) algorithm, also known as the maximum *a posteriori* probability (MAP) algorithm, is useful when designing equalizer and decoder algorithms [1,2]. The BCJR algorithm receives soft probabilistic information regarding the input and produces *a posteriori* probabilistic information regarding the output, and is aptly called a soft-input soft-output (SISO) algorithm. The availability of reliable soft information at the input allows for more accurate *a posteriori* estimates to be produced at the output, which improves overall system performance [3-5].

The equalizer can be designed by using the Viterbi algorithm (VA), also known as the maximum likelihood sequence estimation (MLSE) algorithm, which makes use of the min-sum algorithm to find the most probable transmitted sequence [2,6,7]. The VA performs optimally but it is only able to produce hard estimates at the output, and is therefore not an attractive choice when the equalizer is followed by a SISO decoder. The VA can be modified to produce suboptimal posterior probabilistic information at the output, resulting in the soft output Viterbi algorithm (SOVA) in [8], but because of its suboptimal nature the overall system performance will also be suboptimal. Using the MAP algorithm, the equalizer is able to produce optimal posterior probabilistic information regarding the transmitted information, which can be exploited by the SISO decoder.

While the VA and SOVA can also be used for decoding, the MAP algorithm is the algorithm of choice when the output of the decoder is fed back to be used by the equalizer, a technique known as turbo equalization [3,4]. However, when the estimates of the uncoded transmitted symbols are taken directly from the output of the decoder, i.e., when no turbo equalization is performed, the MLSE algorithm will suffice.

*Correspondence: herman.myburgh@up.ac.za
[1] Department of Electrical, Electronic and Computer Engineering, University of Pretoria, Pretoria 0002, South Africa
Full list of author information is available at the end of the article

Joint equalization and decoding can be performed by MLSE or MAP algorithms and by employing a super-trellis, as shown in [9], given that the depth of the interleaver is limited by computational complexity limitations. The joint equalizer and decoder achieves optimal non-iterative equalization and decoding. Since the input of the joint equalizer and decoder is ISI-corrupted coded transmitted symbols, and the output is the equalized and decoded symbol estimates, no posterior probabilistic information is required at the output. Therefore, the MLSE algorithm can also be used. The MAP algorithm will perform just as well, albeit with approximately twice the computational complexity. It has also been shown in [10] that joint decoding of turbo codes can be done on a super-trellis as well.

Figure 1 shows a block diagram of the communication system considered in this article. The source information is encoded, after which an interleaver is used to separate adjacent coded bits temporally. The coded bits are mapped to modulation symbols chosen from a modulation alphabet $\mathcal{D}$, after which the symbols are used to modulate the carrier before transmission. The transmitted information passes through a multipath white Gaussian noise channel and is received by the receiver antenna. After reception the signal is demodulated and matched filtered in order to produce a received symbol sequence. The CIR is estimated using a number of known pilot symbols, and is provided as input to the equalizer together with the received symbol sequence. The equalizer reverses the effect of the multipath channel and produces optimal symbol estimates (MAP) or an optimal sequence estimate (MLSE) regarding the transmitted coded bits after interleaving. The output of the equalizer is deinterleaved and provided as input to the decoder, which produces optimal estimates regarding the uncoded transmitted information in the form of log-likelihood ratios (LLR), which are mapped back to bits to produce a final estimate of the source information. When joint equalization and decoding is performed, the equalizer, deinterleaver, and decoder are replaced by one functional block, as indicated by the dashed line around these functional blocks. Equalization and decoding are performed simultaneously, producing LLR estimates of the source information at the output.

Equalization, decoding, and joint equalization and decoding will be discussed in the context of the MAP algorithm, but for completeness the MLSE algorithm will also be discussed. The MLSE and MAP algorithms are discussed next, after which these algorithms will subsequently be related to equalization and decoding.

2 The MLSE and MAP algorithms

The MLSE and MAP algorithms are used for equalization as well as for the decoding of convolutional codes [1,2,6,7,11]. The MLSE algorithm is able to optimally estimate the most probable sequence of transmitted symbols/codewords (depending on equalization/decoding), while the MAP algorithm exactly estimates the probability of each transmitted symbol/codeword. These algorithms are underpinned by Bayes' rule of conditional probability, which states that

$$\begin{aligned} rclP(d_t = d^{(m)}|r_t) &= \frac{P(d_t = d^{(m)})P(r_t|d^{(m)})}{P(r_t)} \\ &= \frac{P(d_t = d_t^{(m)})P(r_t|d_t^{(m)})}{\sum_{n=1}^{M} P(d_t = d^{(n)})P(r_t|d_t^{(n)})} \\ &= \beta P(r_t|d^{(m)}) \end{aligned} \tag{1}$$

where $P(d_t = d^{(m)})$ is the prior probability of transmitting symbol/codeword $d^{(m)}$ time instant t. For equalization, the symbols $d^{(m)}$ are chosen from a given modulation

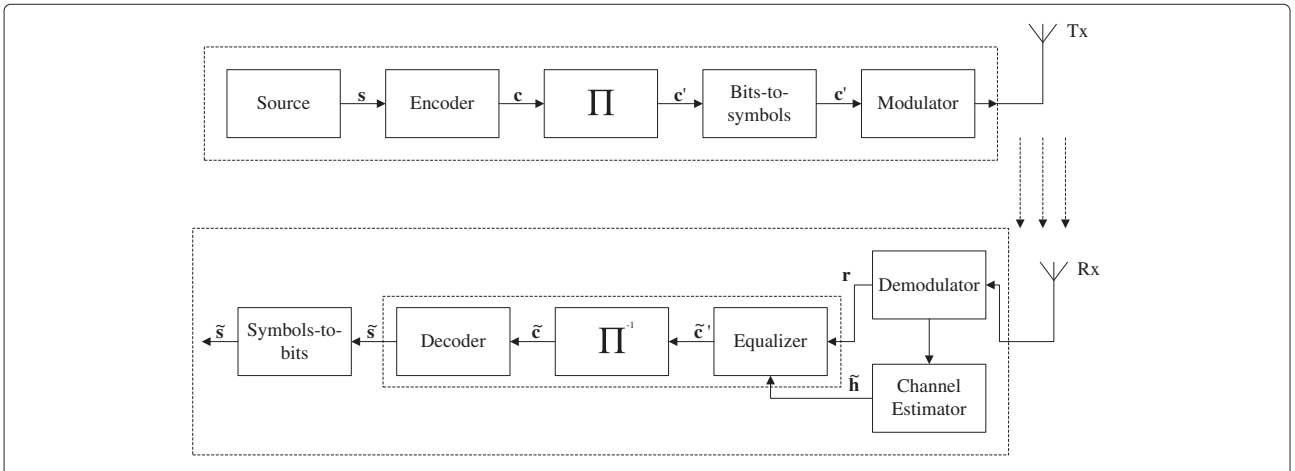

Figure 1 Wireless communication system block diagram.

alphabet $\mathcal{D}$ of size M, where $m = 1, 2, \ldots, M$, and for decoding, codewords symbols $d^{(m)}$ are chosen from a list of M possible codewords. Since it is assumed that the received symbol/codeword sequence is corrupted by white Gaussian noise, the probability of receiving r_t and time instant t having transmitted $d_t^{(m)}$, is expressed as [1,2,6,7,11]

$$P(r_t|d^{(m)}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{|\Delta_t^{(m)}|^2}{2\sigma^2}\right), \tag{2}$$

where σ is the noise standard deviation and $\Delta_t^{(m)}$ is a cost function for the purpose of minimizing the Euclidean distance between the received symbol/codeword and the symbol/codeword to be estimated, which will be discussed in Sections 3 and 4 for equalization and decoding, respectively. Since $P(d_t = d^{(m)})$ and $P(r_t)$ are independent of the choice of $d^{(m)}$, they can be absorbed into a normalization constant β, which, since $\sum_{n=1}^{(m)} P(d_t = d^{(n)}) = 1$, can be expressed as

$$\beta = \frac{1}{\sum_{n=1}^{M} P(d_t = d^{(n)})}. \tag{3}$$

In order to apply Bayes' rule over a transmitted block of N symbols/codewords, the product rule can be used to express (1) for a sequence of length N such that

$$\begin{aligned} P(d_1, d_2, \ldots, d_N|r_1, r_2, \ldots, r_N) &= \beta P(r_1|d_1).P(r_1|d_1).\ldots.P(r_N|d_N) \\ P(\mathbf{d}|\mathbf{r}) &= \beta \prod_{t=1}^{N} P(r_t|d_t) \\ &= \beta \prod_{t=1}^{N} \left(\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\Delta_t^m}{2\sigma^2}\right)\right) \quad , \\ &= \frac{\beta}{(\sqrt{2\pi}\sigma)^N} \exp\left(-\frac{\sum_{t=1}^{N} \Delta_t^m}{2\sigma^2}\right) \\ &= \frac{\beta}{(\sqrt{2\pi}\sigma)^N} \exp\left(-\frac{\Delta}{2\sigma^2}\right) \end{aligned} \tag{4}$$

where any symbol/codeword d_t in the symbol/codeword sequence $\mathbf{d} = \{d_1, d_2, \ldots, d_N\}$ can be substituted for any $d^{(m)}$, and where $\mathbf{r} = \{r_1, r_2, \ldots, r_N\}$ is the received symbol/codeword sequence.

2.1 The MLSE algorithm

In 1972, Forney [11] showed that the VA [6,7,11], first developed by Viterbi in 1967 to decode convolutional error correction codes, can be used to determine the most likely transmitted sequence. This is done by using a trellis, a special graph, or remerging tree structure, representing all possible combinations of transmitted symbols, to determine the solution with the lowest cost through the trellis. The sequence of symbols with the lowest cost maximizes the probability that said sequence was transmitted, thus producing the optimal estimate for the transmitted sequence [2,11].

The VA, or MLSE algorithm, attempts to minimize the cumulative cost function $\Delta = \sum_{t=1}^{N} \Delta_t^m$ in (4), which in turn maximizes $P(\mathbf{d}|\mathbf{r})$ in (4). After minimizing Δ in (4), there is no sequence of transmitted symbols/codewords that is more likely to have been transmitted. However, the probability of each estimated symbol/codeword in the sequence will not necessarily be maximized among all possible transmitted symbols/codewords. As stated before, the MAP algorithm is used to calculate exact posterior probabilities on each symbol/codeword. The min-sum algorithm is used to find the MLSE solution and is discussed next.

2.1.1 The min-sum algorithm

In order to perform optimal sequence estimation, the min-sum algorithm is used. Viterbi [6] and Forney [11] showed that the min-sum algorithm can be used by constructing a trellis and tracing a path, corresponding to the most likely transmitted symbol/codeword sequence through the trellis. The min-sum algorithm allows for the elimination of more costly contending paths at each state on a trellis, thereby greatly reducing the computational complexity due to complete enumeration. For equalization, there will always be M^{L-1} possible paths at every stage in the trellis, where M is the modulation alphabet size and L is the channel memory length. Similarly, for decoding, there will always be 2^{K-1} remaining paths at every stage in the trellis (if convolutional encoding is performed in $GF(2)$), where K is the encoder constraint length which introduces memory to the transmitted codewords. M^{L-1} and 2^{K-1} also correspond to the number of respective states in the trellis for each time instant t, for the equalizer and the decoder.

Figure 2 shows a trellis with $M^{L-1} = 2^{K-1} = 4$ states, corresponding to an equalizer used in a system where a BPSK modulated symbol sequence of length N is transmitted through a multipath channel with a CIR of length $L = 3$, or to a decoder used to decode a coded sequence of N codewords where a convolutional encoder with constraint length $K = 3$ is used. The trellis is initiated by assuming that it starts at state S_1 at time instant $t = 0$ and it is terminated at state S_1 at time instant $t = N$. For equalization, each state represents a unique combination of modulation symbols s^m, where $m = 1, 2, \ldots, M$, from a modulation alphabet $\mathcal{D}$ of size M, and for decoding each state represents a possible codeword at the output of the convolutional decoder. The edges or transitions from state S_i, $i = 1, 2, 3, 4$ at time instant t to any other state S_j, $j = 1, 2, 3, 4$ at time instant $t + 1$ describes the likelihood of this occurrence, which will be a maximum if

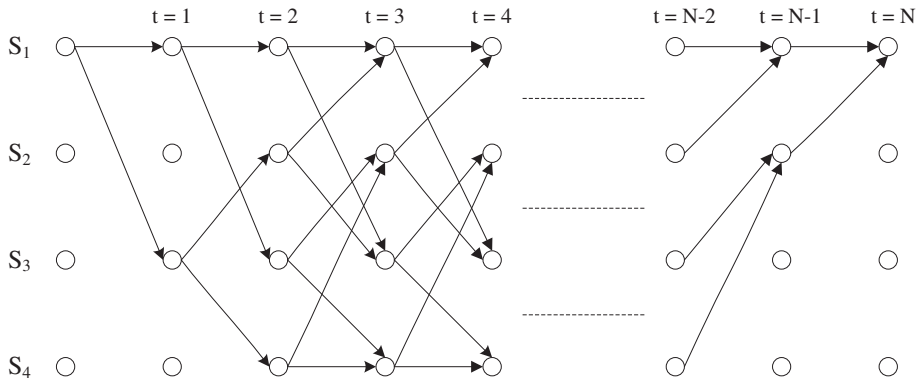

Figure 2 Trellis diagram used to aid the explanation of the equalizer and the decoder.

the cost function $|\Delta_t^{(m)}|^2$ in (2) is a minimum. Table 1 shows the values associated with states $S_j, j = 1, 2, 3, 4$ for equalization and decoding, respectively.

For equalization, each state will have M incoming transitions (from the left), while there will be two incoming transitions for decoding.[a] But as stated above, there will only be $2^{L-1} = 2^{K-1} = 4$ remaining paths at each time instant in the trellis, which means that all but one path has to be eliminated at each state. To eliminate more costly paths at each state in the trellis at time t, the cumulative cost function $\Delta = \sum_{t=1}^{P} \Delta_t^m$ in (4) is calculated for all possible paths leading to the said state up to time t, where the cost of the contending paths are compared, and the path with the highest cost (corresponding to the lowest probability), is eliminated. Therefore, at each state all the previous $\Delta_{t,j\to i}$'s are accumulated, and where there are contending paths, the path with the largest accumulated cost is eliminated.

There will therefore be $2^2 = 4$ surviving paths in each stage on the trellis for stages beyond $t = 2$. When stages $t = N-2$ to $t = N$ are considered, the number of allowed transitions decrease, because of the known tail symbols at the end of the transmitted symbol sequence. For the last few states in the trellis the contending paths are also eliminated, until only one possible path remains at stage $t = N$. This path is then traced back to determine the most probable sequence of transmitted symbols/codewords.

The MLSE equalizer/decoder produces outputs from a set of M symbols for equalization or from a set of uncoded bits for decoding. These estimates are called hard outputs, since the estimates do not contain any probabilistic information regarding the reliability of those estimates. The MAP equalizer can be used to produce probabilistic information as an indication of the reliability of the estimates. The MAP algorithm is discussed next.

Table 1 Equalizer/decoder state values

	S_1	S_2	S_3	S_4
Equalizer	$1, 1$	$-1, 1$	$1, -1$	$-1, -1$
Decoder	$-1, -1$	$1, -1$	$-1, 1$	$1, 1$

2.2 The MAP algorithm

Following the development of the Viterbi MLSE decoding algorithm [6], the BCJR algorithm [1], named after its developers Bahl–Cocke–Jelinek–Raviv, also known as the MAP algorithm, was developed in 1974, also for the decoding of convolutional codes. In the artificial intelligence community, this algorithm was developed independently by Pearl [12] and is called BP. The MAP algorithm is able to produce the posterior probability of each symbol in the estimated transmitted sequence, as opposed to maximizing the probability of the whole transmitted sequence, as done by the Viterbi MLSE equalizer [2,6,7,11].

The aim of the MAP algorithm is to maximize the posterior probability distribution for each transmitted symbol/codeword. While the MLSE algorithm assumes the prior probabilities $P(\mathbf{d})$ of the transmitted symbols to be equal, the MAP algorithm is able to exploit the prior probabilities $P(\mathbf{d})$, if necessary, in order to enhance the quality of posterior probabilistic information on each individual transmitted symbol. Like the MLSE algorithm, the MAP algorithm uses the model in (4) on a trellis, but unlike the MLSE algorithm, the MAP algorithm propagates the transition probabilities forward from past states to future states, as well as backwards from future states to past states, after which the marginalized probability for each estimated symbol/codeword d_t is produced, given past and future information. That is [1,13]

$$P(d_t = d^{(m)}|\mathbf{r}) = \sum_{d_{t^\dagger}:t^\dagger \neq t}^{N} P(\mathbf{d}|\mathbf{r}) \tag{5}$$

where again $d^{(m)}$, $m = 1, 2, \ldots, M$, is the mth symbol chosen from a modulation alphabet $\mathcal{D}$ of size M for equalization, or $d^{(m)}$ is the mth codeword chosen from a list

of M codewords produced by a convolutional encoder for decoding, $\mathbf{r}$ is the received sequence, and N is the number of symbols/codewords in the received sequence.

Referring to Figure 2, the probability of a transition from state $S_{j,t-1}$ (at time $t-1$) to state $S_{i,t}$ (at time t), where $i, j = 1, 2, \ldots, M^{L-1}$ for equalization and $i, j = 1, 2, \ldots, 2^{K-1}$ for decoding, is given by

$$\omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) = \beta P(d_t = d^{(m)}) P(r_t | S_{j,t-1}, S_{i,t}), \quad (6)$$

where β is a normalization constant and $P(r_t|S_{j,t-1}, S_{i,t})$ is given by

$$P(r_t | S_{j,t-1}, S_{i,t}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-|\Delta|^2}{2\sigma^2}\right). \quad (7)$$

By considering the relevant transition value [b] d_ξ, $P(d_t)$ can be written as [3,4]

$$P(d_t = d^{(m)}) = \exp\left(\frac{1}{2} d_\xi L(\tilde{d}_t)\right), \quad (8)$$

where $L(.)$ denotes the LLR operation and $\tilde{d}_t$ is an estimate of d_t. Therefore, substituting (7) and (8) in (6) yields

$$\omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) = \frac{\delta}{\sqrt{2\pi\sigma^2}} \times \exp\left(\frac{-|\Delta|^2}{2\sigma^2} + \exp\left(\frac{1}{2} d_\xi L(\tilde{d}_t)\right)\right), \quad (9)$$

which completely describes the probability of a transition from $S_{j,t-1}$ (or d_j) at time $t-1$ to $S_{i,t}$ (or d_i) at time t, based on all available information. Thus, for a transition labeled $d_\xi = 1$ and $L(\tilde{d}_t)$ equal to any large positive value,[c] $P(d_t = d^{(m)})$ will be large, confirming the transition. Similarly, for a transition labeled $d_\xi = -1$ and $L(\tilde{d}_t)$ equal to any large negative value $P(d_t = d^{(m)})$ will be large, also confirming the transition. However, for a transition labeled $d_\xi = 1$ and $L(d_\xi)$ equal to any large negative number, or for a transition labeled $d_\xi = -1$ and $L(\tilde{d}_t)$ equal to any large positive number, $P(d_t = d^{(m)})$ will be small. Thus, if the prior information $L(\tilde{d}_t)$ is in agreement with the transition value d_ξ, the probability of that transition will increase, confirming that transition. Otherwise, if the prior information $L(\tilde{d}_k)$ contradicts the transition value d_ξ, the probability of that transition will be decreased.

Propagating the transition probabilities $\omega_{j\to i,t}(S_{j,t-1}, S_{i,t})$ across the whole sequence from left to right and from right to left, respectively, and marginalizing according to (5), will produce the posterior probabilities of each estimated d_k. The sum-product algorithm achieves exactly this [13].

2.2.1 The sum-product algorithm

The sum-product algorithm, also known as the forward–backward algorithm, uses the trellis in Figure 2 to perform marginalization. This algorithm follows three steps [13]:

1. Determine the forward pass messages from left to right on the trellis.
2. Determine the backward pass messages from right to left on the trellis.
3. Multiply, scale and accumulate (marginalize) probabilities at each stage of the trellis.

To determine the forward pass messages on the trellis, let a counter t run from left to right (from 1 to N) on the trellis and compute for each state in the trellis

$$\alpha_{i,t} = \sum_{j \epsilon Parent(i)} \omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) \alpha_{j,t-1}$$

where j represents the parent states of the current state at stage i of the trellis and $\omega_t(S_{j,t-1}, S_{i,t})$ is the probability associated with the transition from $S_{j,t-1}$ to $S_{i,t}$. Note that $\alpha_{j,0} = 1$. Similarly, to determine the backward pass messages on the trellis, let a counter t run from right to left (from $N-1$ to 1) on the trellis and compute for each state in the trellis

$$\beta_{j,t} = \sum_{j \epsilon Parent(i)} \omega_{j\to i,t}(S_{j,t}, S_{i,t+1}) \beta_{i,t+1}$$

where again j represents the parent states of the current state at stage i of the trellis and $\omega_t(S_{j,t}, S_{i,t+1})$ is the probability associated with the transition from $S_{j,t}$ to $S_{i,t+1}$. Note that $\beta_{i,t} = 1$. Finally, the exact marginalized symbol probability is determined by summing over all states at each time instant t corresponding to a transition of either $d_\xi = 1$ or $d_\xi = -1$ such that

$$P(d_t = 1|\mathbf{r}) = \sum_{j \epsilon Parent(i), d_\xi = 1} \alpha_{j,t-1} \omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) \beta_{i,t} \quad (10)$$

$$P(d_t = -1|\mathbf{r}) = \sum_{j \epsilon Parent(i), d_\xi = -1} \alpha_{j,t-1} \omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) \beta_{i,t}. \quad (11)$$

The MAP algorithm can also produce soft bits. The soft bits, also called LLRs, can be determined by

$$L(\tilde{s}_t) = \log\left(\frac{P(d_t = 1|\mathbf{r})}{P(d_t = -1|\mathbf{r})}\right), \quad (12)$$

where the sign of $L(\tilde{d}_t)$ indicates whether $\tilde{d}_t = -1$ or $\tilde{d}_t = 1$, and $|L(\tilde{s}_t)|$ is a measure of the confidence of that estimate.

3 Equalization

A mobile communication system transmission channel is characterized by multipath and fading. Multipath is the phenomenon resulting from time spreading of the transmitted signal as it is transmitted through the channel. Fading, on the other hand, results from time variations in the structure of the transmission medium, causing the nature of the multipath channels to vary with time [2]. During transmission, the transmitted symbols pass through the channel, which acts like a filter. The channel has a continuous impulse response, which is estimated at the receiver, to aid in the estimation of the transmitted information.

Each coefficient, or tap, in the impulse response of a multipath fading channel is modeled as a continuous function of time, where each coefficient in the impulse response corresponds to symbol period intervals tT_s. As such, a tapped delay line is used to model the behavior of this channel, as shown in Figure 3. Figure 3 indicates that the tth transmitted symbol s_t is delayed by T_s seconds $L-1$ times, where L is the CIR length and T_s is the symbol period. Each delayed copy of s_t is multiplied by $h_l^{(t)}$, $l = 1, 2, \ldots, L-1$, corresponding to the lth delay branch at time t. Therefore, the tth received symbol can be described by [2,11]

$$r_t = \sum_{l=0}^{L-1} h_t^{(l)} s_{t-l} + n_t, \tag{13}$$

where n_t is the tth noise sample from the distribution $\mathcal{N}(\mu = 0, \sigma^2 = 1)$. Each $h_t^{(l)}$ is a sample taken from one of L time-varying functions, where t corresponds with the tth transmitted symbol and $l = 1, 2, \ldots, L-1$ is the CIR tap number. Each CIR tap is modeled as an independent uncorrelated Rayleigh fading sequence, using the model in [14].

If it is assumed that the CIR is time-invariant for the duration of a data block, (13) can be rewritten as

$$r_t = \sum_{l=0}^{L-1} h_l s_{t-l} + n_t, \tag{14}$$

where s_t denotes the tth complex symbol in the transmitted sequence of N symbols chosen from an alphabet $\mathcal{D}$ containing M complex symbols, r_t is the tth received symbol, n_t is the tth noise sample from the distribution $\mathcal{N}(\mu = 0, \sigma^2 = 1)$, and h_l is the lth coefficient of the estimated CIR. Equalization is performed under the assumption that each CIR coefficient h_l is time-invariant for the duration of a data block. The CIR $\mathbf{h} = \{h_0, h_1, \ldots, h_{L-1}\}$ therefore completely describes the multipath channel for a given received data block, assuming that the data block is sufficiently short so as to render the CIR time-invariant. The equalizer takes as input the received symbol sequence $\mathbf{r}$ as well as the CIR $\mathbf{h}$.

To estimate the transmitted sequence of length N optimally in a wireless communication system transmitting modulated symbols though a multipath channel, the cumulative cost function in (4)

$$\begin{aligned} rcl\Delta &= \sum_{t=1}^{N} \Delta_t^m \\ &= \sum_{t=1}^{N} |r_t - \sum_{l=0}^{L-1} h_l s_{t-l}|^2 \end{aligned} \tag{15}$$

must be minimized [2,11]. Here $\mathbf{s} = \{s_1, s_2, \ldots, s_N\}$ is the most likely transmitted sequence that will maximize (4). Although the minimization of (15) will maximize (4), the resulting sequence estimates will only be optimal in the sequence sense. Depending on the application and whether the equalizer is followed by a decoder, either optimal sequence estimation or exact posterior probabilistic symbol estimation can be performed using the MLSE or MAP algorithms discussed above.

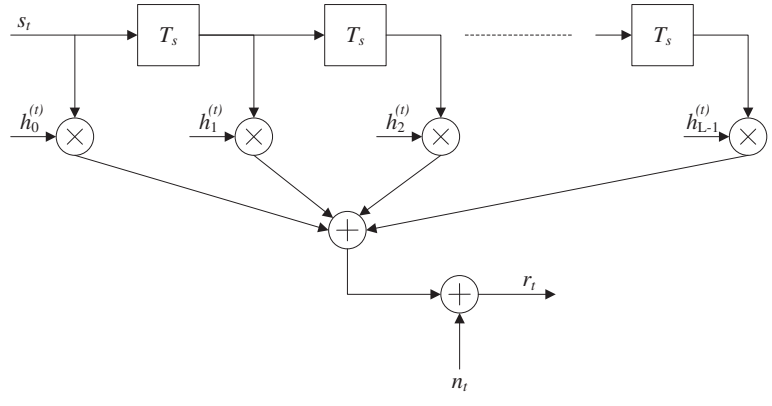

Figure 3 Tapped delay line used to simulate ISI in a multipath fading channel.

4 Decoding

In a mobile communication system, the transmitted signal is subjected to energy losses due to multipath and fading as well as interference due to thermal noise, resulting in unreliable estimates of source information in the receiver. In order to correct errors in the receiver, error control coding (ECC) is used to introduce controlled redundancy to the source information. The decoder exploits the structure introduced to the encoded symbol sequence by the encoder, to reconstruct the source information from the estimated coded symbols, correcting errors while doing so.

ECC plays an important role in digital communication systems. As the name suggests, ECC is used to allow for the correction of errors in the received symbol sequence. ECC is performed by adding controlled redundancy to the information being transmitted. Since the redundant information is mathematically related to the original information, errors can be corrected [2]. Although the redundancy adds an overhead to the transmitted data, the performance increase overshadows this drawback. During the last few decades, ECC has been the subject of much research and significant contributions and advancements have been made.

The coding scheme considered in this article is convolutional encoding, as it is most often used in turbo equalization. A convolutional code is generated by passing the information sequence to be transmitted through a linear finite state shift register, consisting of K stages.[d] The binary input is shifted into the shift register k bits at a time, producing n output bits. The code rate is therefore $R_c = k/n$. The encoder can be expressed in terms of n generator polynomials, describing the connections between the states of the encoder shift register. Figure 4 shows the rate $R_c = k/n = 1/3$, constraint length $K = 3$, convolutional encoder considered in this article.

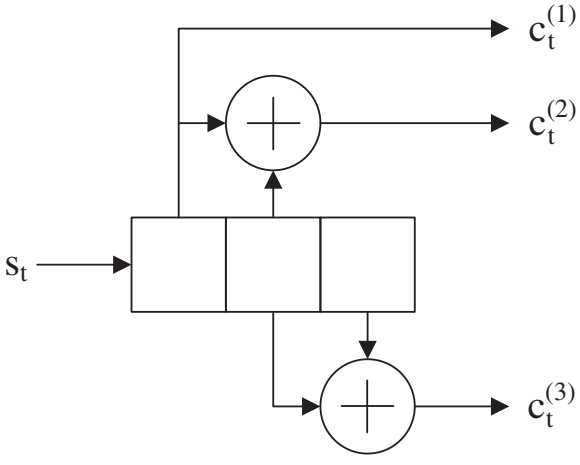

Figure 4 Rate $1/3$ convolutional encoder used in this article.

The generator polynomials that describe the connections between the elements in the shift register are given in sequence form as $g_1 = [100], g_2 = [110]$, and $g_3 = [011]$, which can also be written in octal form as $G = [4, 6, 3]$ or in polynomial form as $G = [1, 1 + X, X + X^2]$. For every input source bit s_t the encoder produces $k = 3$ output bits $c_t^{(1)}$, $c_t^{(2)}$, and $c_t^{(3)}$, where

$$\begin{aligned} c_t^{(1)} &= s_t \\ c_t^{(2)} &= s_t \oplus s_{(t-1)} \\ c_t^{(3)} &= s_{(t-1)} \oplus s_{(t-2)}, \end{aligned} \tag{16}$$

where $\oplus$ is the exclusive OR operation. It is assumed that the encoder starts in the all-zero state for every data block that is encoded. Also, the encoder is forced into a zero-state after the data block is encoded by appending $K - 1$ zero bits to the data block. This is done to enable decoding using an MLSE or a MAP decoder [1,6,7].

Each convolution encoder has a corresponding state diagram, which is used to map state transitions to encoder outputs, which in turn are used to determine the most likely state transitions during decoding. The state diagram of the encoder in Figure 4 is shown in Figure 5. Each state contains two bits representing the two leftmost elements in the encoder shift register. As bits are shifted through the encoder $k = 1$ bit at a time, transitions occur, with each state transition producing $n = 3$ output bits. The dashed lines are associated with transitions resulting from a zero at the input of the encoder, and solid lines are association with a one at the input of the encoder.

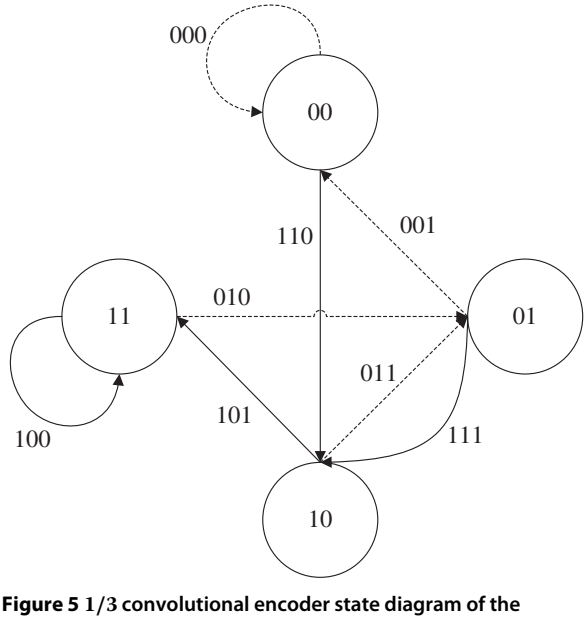

Figure 5 $1/3$ convolutional encoder state diagram of the convolutional encoder in Figure 4.

The decoding of convolutional codes is closely related to equalization discussed in Section 3, in that the min-sum (Viterbi MLSE) and sum-product (MAP/BCJR) algorithms can also be used for decoding. The MAP algorithm is an attractive choice for use in iterative equalization/decoding algorithms. It is attractive for two reasons: first because it includes prior probabilistic information in the estimation, and second, because it provides soft posterior estimates regarding individual coded or uncoded symbols, which in turn can be used as prior information in subsequent iterations.

To decode a convolutional code using the MLSE or MAP algorithms discussed above, the cumulative cost function in (4)

$$\begin{aligned} rcl\Delta &= \sum_{t=1}^{N} \Delta_t^m \\ &= \sum_{t=1}^{N} \sum_{n=1}^{k} |r_t^{(n)} - c_t^{(n)}|^2 \end{aligned} \tag{17}$$

is minimized [6,11], where $\mathbf{r}_t = \{r_t^{(1)}, \ldots, r_t^{(n)}\}$ are the received coded symbols corresponding to codeword $\mathbf{c}_t = \{c_t^{(1)}, \ldots, c_t^{(k)}\}$ at time instant t, selected based on the encoder output generated by the relevant state transition.[e] Here, k is the number of output bits generated by the encoder. The MLSE or MAP algorithms can be applied to find either the most probable transmitted sequence of codewords, or the most probable transmitted uncoded and/or coded symbols. The output of the latter can be used in a turbo equalizer as feedback to aid the equalizer in making more informed decisions.

5 Non-iterative joint equalization and decoding (NI-JED)

In conventional single-carrier wireless communication systems, where the coded information is transmitted through a multipath channel, equalization and decoding, as explained in Section 3 and 4, are performed separately. However, joint equalization and decoding can also be performed on a single trellis when it is assumed that the coded symbols are not interleaved before transmission, or that certain assumptions are made regarding the structure of the interleavers used to interleave the coded symbols.

Interleavers allow for the temporal separation of adjacent coded symbols during transmission, so that when burst errors occur in response to a loss of signal power during fading, the resulting errors in an error burst will be distributed throughout the data block after deinterleaving. This leads to performance gains in the receiver, as the equalizer and the decoder are able to infer information and correct single errors more accurately, as opposed to correcting consecutive, or burst errors. Therefore, interleavers are employed in a wireless communication system where the transmitted signal is subject to multipath and fading [2].

In [9], an optimal joint equalizer and decoder is presented, where the equalizer and the decoder are jointly modeled on one trellis, a super-trellis according to [9], and a block interleaver is used to interleave the coded symbols before transmission. The computational complexity of this joint equalizer and decoder, however, is not only exponentially related to the channel memory length and the encoder constraint length, but also to the interleaver depth. The interleaver depth determines the degree of separation between the coded symbols, which has a direct effect on system performance. In order to improve system performance the interleaver depth has to be increased, which results in an exponential increase in computational complexity. Ideally a random interleaver should be used for maximum performance gains, but a random interleaver has a devastating effect on the causality relationship between transmitted and deinterleaved received symbols. This joint equalizer/decoder is therefore only feasible when limited depth block interleavers are used. Joint equalization and decoding using a super-trellis is discussed next, first for systems where no interleaving is performed, and then for systems where depth-limited interleavers are used.

5.1 No interleaving

Noting the striking similarities between MAP equalization and MAP decoding in Section 2, a possible joint equalization and decoding algorithm can be envisioned. In order to equalize and decode a received symbol sequence jointly, the cumulative costs for equalization in (15) and decoding in (17) must be combined in order to equalize while decoding. Either the MLSE or MAP algorithms can be used for this purpose, since the super-trellis encapsulates all the available information and therefore no exchange of information between equalizer and decoder is necessary. No prior information is therefore required when calculating state transitions on a super-trellis.

Consider a system transmitting non-interleaved coded information through a multipath channel of length L. A rate $R_c = k/n$ constraint length K convolutional encoder is used to produce a sequence $\mathbf{c}$ of coded bits of length N_c from an uncoded bit sequence $\mathbf{s}$ of length N_u. The number of super-trellis states required to perform joint equalization and decoding is the product of the number of states of the individual trellises required to perform equalization and decoding separately. Therefore, the number of super-trellis states is $M^{L-1}M^{K-1} = M^{(L-1)+(K-1)}$, where $M = 2$ due to BPSK modulation as well as $GF(2)$ encoding.

When the MAP equalizer was considered, the probability of a transition from uncoded *symbol* s_j transmitted and time $t-1$ to s_i transmitted at time t

was calculated (see (6)), and when the MAP decoder was considered, the probability of a transition from *codeword* $\mathbf{c}_j$ transmitted at time $t-1$ to $\mathbf{c}_i$ transmitted at time t was calculated (although d_j and d_i were used instead for both the equalization of symbols and decoding of codewords). When modeling the equalizer and decoder jointly, however, transitions between super states are calculated on a super-trellis, where each state represents the uncoded symbols that constitute a codeword, together with interfering symbols of previous codewords due to multipath.

Suppose a communication system encodes a source bit sequence $\mathbf{s}$ of length N_u with a rate $R_c = k/n = 1/3$, constraint length $K = 3$, convolutional encoder with arbitrary generator polynomials $g_1 = \{g_1^{(1)}, g_1^{(2)}, g_1^{(3)}\}$, $g_2 = \{g_2^{(1)}, g_2^{(2)}, g_2^{(3)}\}$ and $g_3 = \{g_3^{(1)}, g_3^{(2)}, g_3^{(3)}\}$, in order to produce a coded bit sequence $\mathbf{c}$ of length $N_c = N_u/R_c$, which is BPSK modulated and transmitted through a multipath channel with a CIR $\mathbf{h}$ of length L. The encoder produces the coded bit sequence

$$\mathbf{c} = \Big\{ c_1^{(1)}, c_1^{(2)}, c_1^{(3)}, c_2^{(1)}, c_2^{(2)}, c_2^{(3)}, \ldots, c_{N_u-1}^{(1)}, c_{N_u-1}^{(2)}, c_{N_u-1}^{(3)}, c_{N_u}^{(1)}, c_{N_u}^{(2)}, c_{N_u}^{(3)} \Big\}$$

of length N_c from the length N_c uncoded bit sequence

$$\mathbf{s} = \{s_1, s_2 \ldots, s_{N_u-1}, s_{N_u}\}.$$

5.1.1 Channel length $2 \leq L < 5$

When the coded sequence is transmitted through a channel with lengths $L = 2$, $L = 3$, and $L = 4$, the respective resulting received sequences, without noise, can expressed in terms of the coded symbols

$$\begin{aligned} r_t^{(1)} &= c_t^{(1)} h_0 + c_{t-1}^{(3)} h_1 \\ r_t^{(2)} &= c_t^{(2)} h_0 + c_t^{(1)} h_1 \\ r_t^{(3)} &= c_t^{(3)} h_0 + c_t^{(2)} h_1 \end{aligned}, \tag{18}$$

$$\begin{aligned} r_t^{(1)} &= c_t^{(1)} h_0 + c_{t-1}^{(3)} h_1 + c_{t-1}^{(2)} h_2 \\ r_t^{(2)} &= c_t^{(2)} h_0 + c_t^{(1)} h_1 + c_{t-1}^{(3)} h_2 \\ r_t^{(3)} &= c_t^{(3)} h_0 + c_t^{(2)} h_1 + c_t^{(1)} h_3 \end{aligned}, \tag{19}$$

and

$$\begin{aligned} r_t^{(1)} &= c_t^{(1)} h_0 + c_{t-1}^{(3)} h_1 + c_{t-1}^{(2)} h_2 + c_{t-1}^{(1)} h_3 \\ r_t^{(2)} &= c_t^{(2)} h_0 + c_t^{(1)} h_1 + c_{t-1}^{(3)} h_2 + c_{t-1}^{(2)} h_3 \\ r_t^{(3)} &= c_t^{(3)} h_0 + c_t^{(2)} h_1 + c_t^{(1)} h_3 + c_{t-1}^{(3)} h_3 \end{aligned}, \tag{20}$$

which can, respectively, be expressed in terms of the uncoded symbols

$$\begin{aligned} r_t^{(1)} &= (g_1^{(1)} s_t \oplus g_1^{(2)} s_{t-1} \oplus g_1^{(3)} s_{t-2}) h_0 \\ &\quad + (g_3^{(1)} s_{t-1} \oplus g_3^{(2)} s_{t-2} \oplus g_3^{(3)} s_{t-3}) h_1 \\ r_t^{(2)} &= (g_2^{(1)} s_t \oplus g_2^{(2)} s_{t-1} \oplus g_2^{(3)} s_{t-2}) h_0 \\ &\quad + (g_1^{(1)} s_t \oplus g_1^{(2)} s_{t-1} \oplus g_1^{(3)} s_{t-2}) h_1 \\ r_t^{(3)} &= (g_3^{(1)} s_t \oplus g_3^{(2)} s_{t-1} \oplus g_3^{(3)} s_{t-2}) h_0 \\ &\quad + (g_2^{(1)} s_t \oplus g_2^{(2)} s_{t-1} \oplus g_2^{(3)} s_{t-2}) h_1 \end{aligned}, \tag{21}$$

for $L = 2$,

$$\begin{aligned} r_t^{(1)} &= (g_1^{(1)} s_t \oplus g_1^{(2)} s_{t-1} \oplus g_1^{(3)} s_{t-2}) h_0 \\ &\quad + (g_3^{(1)} s_{t-1} \oplus g_3^{(2)} s_{t-2} \oplus g_3^{(3)} s_{t-3}) h_1 \\ &\quad + (g_2^{(1)} s_{t-1} \oplus g_2^{(2)} s_{t-2} \oplus g_2^{(3)} s_{t-3}) h_2 \\ r_t^{(2)} &= (g_2^{(1)} s_t \oplus g_2^{(2)} s_{t-1} \oplus g_2^{(3)} s_{t-2}) h_0 \\ &\quad + (g_1^{(1)} s_t \oplus g_1^{(2)} s_{t-1} \oplus g_1^{(3)} s_{t-2}) h_1 \\ &\quad + (g_3^{(1)} s_{t-1} \oplus g_3^{(2)} s_{t-2} \oplus g_3^{(3)} s_{t-3}) h_2 \\ r_t^{(3)} &= (g_3^{(1)} s_t \oplus g_3^{(2)} s_{t-1} \oplus g_3^{(3)} s_{t-2}) h_0 \\ &\quad + (g_2^{(1)} s_t \oplus g_2^{(2)} s_{t-1} \oplus g_2^{(3)} s_{t-2}) h_1 \\ &\quad + (g_1^{(1)} s_t \oplus g_1^{(2)} s_{t-1} \oplus g_1^{(3)} s_{t-2}) h_2 \end{aligned}, \tag{22}$$

for $L = 3$, and

$$\begin{aligned} r_t^{(1)} &= (g_1^{(1)} s_t \oplus g_1^{(2)} s_{t-1} \oplus g_1^{(3)} s_{t-2}) h_0 \\ &\quad + (g_3^{(1)} s_{t-1} \oplus g_3^{(2)} s_{t-2} \oplus g_3^{(3)} s_{t-3}) h_1 \\ &\quad + (g_2^{(1)} s_{t-1} \oplus g_2^{(2)} s_{t-2} \oplus g_2^{(3)} s_{t-3}) h_2 \\ &\quad + (g_1^{(1)} s_{t-1} \oplus g_1^{(2)} s_{t-2} \oplus g_1^{(3)} s_{t-3}) h_3 \\ r_t^{(2)} &= (g_2^{(1)} s_t \oplus g_2^{(2)} s_{t-1} \oplus g_2^{(3)} s_{t-2}) h_0 \\ &\quad + (g_1^{(1)} s_t \oplus g_1^{(2)} s_{t-1} \oplus g_1^{(3)} s_{t-2}) h_1 \\ &\quad + (g_3^{(1)} s_{t-1} \oplus g_3^{(2)} s_{t-2} \oplus g_3^{(3)} s_{t-3}) h_2 \\ &\quad + (g_2^{(1)} s_{t-1} \oplus g_2^{(2)} s_{t-2} \oplus g_2^{(3)} s_{t-3}) h_3 \\ r_t^{(3)} &= (g_3^{(1)} s_t \oplus g_3^{(2)} s_{t-1} \oplus g_3^{(3)} s_{t-2}) h_0 \\ &\quad + (g_2^{(1)} s_t \oplus g_2^{(2)} s_{t-1} \oplus g_2^{(3)} s_{t-2}) h_1 \\ &\quad + (g_1^{(1)} s_t \oplus g_1^{(2)} s_{t-1} \oplus g_1^{(3)} s_{t-2}) h_2 \\ &\quad + (g_3^{(1)} s_{t-1} \oplus g_3^{(2)} s_{t-2} \oplus g_3^{(3)} s_{t-3}) h_3 \end{aligned}, \tag{23}$$

for $L = 4$. It is clear from (21) to (23) that each $\mathbf{r}_t = \{r_t^{(1)}, r_t^{(2)}, r_t^{(3)}\}$ contains uncoded symbols $\{s_t, s_{t-1}, s_{t-2}, s_{t-3}\}$. The transition probability between superstate $S_{j,t-1}$ and $S_{i,t}$ can therefore be calculated, where $S_{i,t}$ represents $\{s_t, s_{t-1}, s_{t-2}\}$ and $S_{j,t-1}$ represents $\{s_{t-1}, s_{t-2}, s_{t-3}\}$. The number of super-trellis states will therefore be $M = 2^3$, as three symbols need to be

represented by each superstate. In order to express the number of states for $2 \leq L < 5$ this author defines

$$Q = 1, \quad \text{if } 2 \leq L \leq 4. \tag{24}$$

The number of super-trellis states can therefore be expressed as $M = 2^{Q+(K-1)} = 2^{(1)+(2)} = 2^3$.

5.1.2 Channel length $5 \leq L < 8$

Transmitting the coded sequence

$$\mathbf{c} = \left\{ c_1^{(1)}, c_1^{(2)}, c_1^{(3)}, c_2^{(1)}, c_2^{(2)}, c_2^{(3)}, \ldots, c_{N_u-1}^{(1)}, c_{N_u-1}^{(2)}, c_{N_u-1}^{(3)}, c_{N_u}^{(1)}, c_{N_u}^{(2)}, c_{N_u}^{(3)} \right\}$$

through a channel length $L = 5$, $L = 6$, and $L = 7$, respectively, the received symbol sequences without noise can be expressed as

$$\begin{aligned} r_t^{(1)} &= c_t^{(1)}h_0 + c_{t-1}^{(3)}h_1 + c_{t-1}^{(2)}h_2 \\ &\quad + c_{t-1}^{(1)}h_3 + c_{t-2}^{(3)}h_4 \\ r_t^{(2)} &= c_t^{(2)}h_0 + c_t^{(1)}h_1 + c_{t-1}^{(3)}h_2 \\ &\quad + c_{t-1}^{(2)}h_3 + c_{t-1}^{(1)}h_4 \\ r_t^{(3)} &= c_t^{(3)}h_0 + c_t^{(2)}h_1 + c_t^{(1)}h_3 \\ &\quad + c_{t-1}^{(3)}h_3 + c_{t-1}^{(2)}h_4 \end{aligned}, \tag{25}$$

$$\begin{aligned} r_t^{(1)} &= c_t^{(1)}h_0 + c_{t-1}^{(3)}h_1 + c_{t-1}^{(2)}h_2 + c_{t-1}^{(1)}h_3 \\ &\quad + c_{t-2}^{(3)}h_4 + c_{t-2}^{(2)}h_5 \\ r_t^{(2)} &= c_t^{(2)}h_0 + c_t^{(1)}h_1 + c_{t-1}^{(3)}h_2 + c_{t-1}^{(2)}h_3 \\ &\quad + c_{t-1}^{(1)}h_4 + c_{t-2}^{(3)}h_5 \\ r_t^{(3)} &= c_t^{(3)}h_0 + c_t^{(2)}h_1 + c_t^{(1)}h_3 + c_{t-1}^{(3)}h_3 \\ &\quad + c_{t-1}^{(2)}h_4 + c_{t-1}^{(1)}h_5 \end{aligned}, \tag{26}$$

and

$$\begin{aligned} r_t^{(1)} &= c_t^{(1)}h_0 + c_{t-1}^{(3)}h_1 + c_{t-1}^{(2)}h_2 + c_{t-1}^{(1)}h_3 + c_{t-2}^{(3)}h_4 \\ &\quad + c_{t-2}^{(2)}h_5 + c_{t-2}^{(1)}h_6 \\ r_t^{(2)} &= c_t^{(2)}h_0 + c_t^{(1)}h_1 + c_{t-1}^{(3)}h_2 + c_{t-1}^{(2)}h_3 + c_{t-1}^{(1)}h_4 \\ &\quad + c_{t-2}^{(3)}h_5 + c_{t-2}^{(2)}h_6 \\ r_t^{(3)} &= c_t^{(3)}h_0 + c_t^{(2)}h_1 + c_t^{(1)}h_3 + c_{t-1}^{(3)}h_3 + c_{t-1}^{(2)}h_4 \\ &\quad + c_{t-1}^{(1)}h_5 + c_{t-2}^{(3)}h_6 \end{aligned}. \tag{27}$$

As before, expressing $\mathbf{r}_t = \{r_t^{(1)}, r_t^{(2)}, r_t^{(3)}\}$ above in terms of the uncoded symbols, one gets

$$\begin{aligned} r_t^{(1)} &= (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_0 \\ &\quad + (g_3^{(1)}s_{t-1} \oplus g_3^{(2)}s_{t-2} \oplus g_3^{(3)}s_{t-3})h_1 \\ &\quad + (g_2^{(1)}s_{t-1} \oplus g_2^{(2)}s_{t-2} \oplus g_2^{(3)}s_{t-3})h_2 \\ &\quad + (g_1^{(1)}s_{t-1} \oplus g_1^{(2)}s_{t-2} \oplus g_1^{(3)}s_{t-3})h_3 \\ &\quad + (g_3^{(1)}s_{t-2} \oplus g_3^{(2)}s_{t-3} \oplus g_3^{(3)}s_{t-4})h_4 \\ r_t^{(2)} &= (g_2^{(1)}s_t \oplus g_2^{(2)}s_{t-1} \oplus g_2^{(3)}s_{t-2})h_0 \\ &\quad + (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_1 \\ &\quad + (g_3^{(1)}s_{t-1} \oplus g_3^{(2)}s_{t-2} \oplus g_3^{(3)}s_{t-3})h_2, \\ &\quad + (g_2^{(1)}s_{t-1} \oplus g_2^{(2)}s_{t-2} \oplus g_2^{(3)}s_{t-3})h_3 \\ &\quad + (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_4 \\ r_t^{(3)} &= (g_3^{(1)}s_t \oplus g_3^{(2)}s_{t-1} \oplus g_3^{(3)}s_{t-2})h_0 \\ &\quad + (g_2^{(1)}s_t \oplus g_2^{(2)}s_{t-1} \oplus g_2^{(3)}s_{t-2})h_1 \\ &\quad + (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_2 \\ &\quad + (g_3^{(1)}s_{t-1} \oplus g_3^{(2)}s_{t-2} \oplus g_3^{(3)}s_{t-3})h_3 \\ &\quad + (g_2^{(1)}s_{t-1} \oplus g_2^{(2)}s_{t-2} \oplus g_2^{(3)}s_{t-3})h_4 \end{aligned} \tag{28}$$

for $L = 5$,

$$\begin{aligned} r_t^{(1)} &= (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_0 \\ &\quad + (g_3^{(1)}s_{t-1} \oplus g_3^{(2)}s_{t-2} \oplus g_3^{(3)}s_{t-3})h_1 \\ &\quad + (g_2^{(1)}s_{t-1} \oplus g_2^{(2)}s_{t-2} \oplus g_2^{(3)}s_{t-3})h_2 \\ &\quad + (g_1^{(1)}s_{t-1} \oplus g_1^{(2)}s_{t-2} \oplus g_1^{(3)}s_{t-3})h_3 \\ &\quad + (g_3^{(1)}s_{t-2} \oplus g_3^{(2)}s_{t-3} \oplus g_3^{(3)}s_{t-4})h_4 \\ &\quad + (g_2^{(1)}s_{t-2} \oplus g_2^{(2)}s_{t-3} \oplus g_2^{(3)}s_{t-4})h_5 \\ r_t^{(2)} &= (g_2^{(1)}s_t \oplus g_2^{(2)}s_{t-1} \oplus g_2^{(3)}s_{t-2})h_0 \\ &\quad + (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_1 \\ &\quad + (g_3^{(1)}s_{t-1} \oplus g_3^{(2)}s_{t-2} \oplus g_3^{(3)}s_{t-3})h_2 \\ &\quad + (g_2^{(1)}s_{t-1} \oplus g_2^{(2)}s_{t-2} \oplus g_2^{(3)}s_{t-3})h_3 \\ &\quad + (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_4 \\ &\quad + (g_3^{(1)}s_{t-2} \oplus g_3^{(2)}s_{t-3} \oplus g_3^{(3)}s_{t-4})h_5 \\ r_t^{(3)} &= (g_3^{(1)}s_t \oplus g_3^{(2)}s_{t-1} \oplus g_3^{(3)}s_{t-2})h_0 \\ &\quad + (g_2^{(1)}s_t \oplus g_2^{(2)}s_{t-1} \oplus g_2^{(3)}s_{t-2})h_1 \\ &\quad + (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_2 \\ &\quad + (g_3^{(1)}s_{t-1} \oplus g_3^{(2)}s_{t-2} \oplus g_3^{(3)}s_{t-3})h_3 \\ &\quad + (g_2^{(1)}s_{t-1} \oplus g_2^{(2)}s_{t-2} \oplus g_2^{(3)}s_{t-3})h_4 \\ &\quad + (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_5 \end{aligned}, \tag{29}$$

for $L = 6$,

$$\begin{aligned}
r_t^{(1)} &= (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_0 \\
&\quad + (g_3^{(1)}s_{t-1} \oplus g_3^{(2)}s_{t-2} \oplus g_3^{(3)}s_{t-3})h_1 \\
&\quad + (g_2^{(1)}s_{t-1} \oplus g_2^{(2)}s_{t-2} \oplus g_2^{(3)}s_{t-3})h_2 \\
&\quad + (g_1^{(1)}s_{t-1} \oplus g_1^{(2)}s_{t-2} \oplus g_1^{(3)}s_{t-3})h_3 \\
&\quad + (g_3^{(1)}s_{t-2} \oplus g_3^{(2)}s_{t-3} \oplus g_3^{(3)}s_{t-4})h_4 \\
&\quad + (g_2^{(1)}s_{t-2} \oplus g_2^{(2)}s_{t-3} \oplus g_2^{(3)}s_{t-4})h_5 \\
&\quad + (g_1^{(1)}s_{t-2} \oplus g_1^{(2)}s_{t-3} \oplus g_1^{(3)}s_{t-4})h_6 \\
r_t^{(2)} &= (g_2^{(1)}s_t \oplus g_2^{(2)}s_{t-1} \oplus g_2^{(3)}s_{t-2})h_0 \\
&\quad + (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_1 \\
&\quad + (g_3^{(1)}s_{t-1} \oplus g_3^{(2)}s_{t-2} \oplus g_3^{(3)}s_{t-3})h_2 \\
&\quad + (g_2^{(1)}s_{t-1} \oplus g_2^{(2)}s_{t-2} \oplus g_2^{(3)}s_{t-3})h_3, \\
&\quad + (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_4 \\
&\quad + (g_3^{(1)}s_{t-2} \oplus g_3^{(2)}s_{t-3} \oplus g_3^{(3)}s_{t-4})h_5 \\
&\quad + (g_2^{(1)}s_{t-2} \oplus g_2^{(2)}s_{t-3} \oplus g_2^{(3)}s_{t-4})h_6 \\
r_t^{(3)} &= (g_3^{(1)}s_t \oplus g_3^{(2)}s_{t-1} \oplus g_3^{(3)}s_{t-2})h_0 \\
&\quad + (g_2^{(1)}s_t \oplus g_2^{(2)}s_{t-1} \oplus g_2^{(3)}s_{t-2})h_1 \\
&\quad + (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_2 \\
&\quad + (g_3^{(1)}s_{t-1} \oplus g_3^{(2)}s_{t-2} \oplus g_3^{(3)}s_{t-3})h_3 \\
&\quad + (g_2^{(1)}s_{t-1} \oplus g_2^{(2)}s_{t-2} \oplus g_2^{(3)}s_{t-3})h_4 \\
&\quad + (g_1^{(1)}s_t \oplus g_1^{(2)}s_{t-1} \oplus g_1^{(3)}s_{t-2})h_5 \\
&\quad + (g_3^{(1)}s_{t-2} \oplus g_3^{(2)}s_{t-3} \oplus g_3^{(3)}s_{t-4})h_6
\end{aligned} \tag{30}$$

for $L = 7$. It is clear from (28) to (30) that for $5 \leq L < 8$ $\mathbf{r}_t = \{r_t^{(1)}, r_t^{(2)}, r_t^{(3)}\}$ contains $\{s_t, s_{t-1}, s_{t-2}, s_{t-3}, s_{t-4}\}$, necessitating super-trellis state $S_{j,t-1}$ and $S_{i,t}$ to represent, respectively, $\{s_{t-1}, s_{t-2}, s_{t-3}, s_{t-4}\}$ and $\{s_t, s_{t-1}, s_{t-2}, s_{t-3}\}$ in order to calculate the transition probability. With the new information, this author redefines Q in (24) as

$$Q = \begin{cases} 1 & \text{if } 2 \leq L < 5 \\ 2 & \text{if } 5 \leq L < 8 \end{cases}.$$

The number of super-trellis states required for $5 \leq L < 8$ is therefore $M = 2^{Q+(K-1)} = 2^{(2)+(2)} = 2^4$.

5.1.3 Channel length L

Given any channel memory length L, encoder constraint length K, code rate $R_c = k/n = 1/3$, the number of bits needed to be represented by a superstate can be determined by $Q + (K-1)$, where

$$Q = \begin{cases} 1 & \text{if } n-1 \leq L < 2(n-1) \\ 2 & \text{if } 2(n-1) \leq L < 3(n-1) \\ 3 & \text{if } 3(n-1) \leq L < 4(n-1) \\ \vdots & \vdots \quad\quad \vdots \\ q-1 & \text{if } (q-1)(n-1) \leq L < (q-2)(n-1) \\ q & \text{if } q(n-1) \leq L < (q-1)(n-1) \end{cases},$$

which in turn can be used to determine the number of super-trellis states by $M = 2^{Q+(K+1)}$. Keeping track of the ISI undergone by the coded symbols due to multipath, and replacing those symbols with their corresponding uncoded symbols, a model for the received symbols can be derived.

The probability of a transition from superstate $S_{j,t-1}$ superstate $S_{i,t}$ is given by

$$\omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) = P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)} | S_{j,t-1}, S_{i,t}) P(s_t), \tag{31}$$

which can be rewritten in terms of the uncoded symbols represented by each state

$$\begin{aligned}
&\omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) \\
&= P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)} | s_t, s_{t-1}, \ldots, s_{t-\log_2(M)}) P(s_t),
\end{aligned} \tag{32}$$

where

$$\begin{aligned}
&P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)} | s_t, s_{t-1}, \ldots, s_{t-\log_2(M)}) \\
&= \frac{1}{\sqrt{2\pi}\sigma} \left(\frac{-\sum_{i=1}^{k} \Delta_{i,t}}{2\sigma^2} \right).
\end{aligned} \tag{33}$$

$\Delta_{i,t}$ can be determined by minimizing the receiver equations such that

$$\begin{aligned}
\Delta_{1,t} &= \left| r_t^{(1)} - \sum_{l=0}^{L-1} h_l (g_u^{(1)} s_{t-m} \oplus g_u^{(2)} s_{(t-m)-1} \oplus g_u^{(3)} s_{(t-m)-2}) \right|^2 \\
\Delta_{2,t} &= \left| r_t^{(2)} - \sum_{l=0}^{L-1} h_l (g_v^{(1)} s_{t-n} \oplus g_v^{(2)} s_{(t-n)-1} \oplus g_v^{(3)} s_{(t-n)-2}) \right|^2, \\
\Delta_{3,t} &= \left| r_t^{(3)} - \sum_{l=0}^{L-1} h_l (g_w^{(1)} s_{t-o} \oplus g_w^{(2)} s_{(t-o)-1} \oplus g_w^{(3)} s_{(t-o)-2}) \right|^2
\end{aligned} \tag{34}$$

where

$$u = \begin{cases} 1 \text{ if } & l = 0, n, 2n, \ldots \\ 3 \text{ if } & l = 1, n+1, 2n+1, \ldots \\ 2 \text{ if } & l = 2, n+2, 3n+1, \ldots \end{cases},$$

and $v = ((u+1) \mod 3)+1$ and $w = ((u+2) \mod 3)+1$. Also m, n and o starts at 0 and increases by 1 each time uncoded symbols from the previous time instant are used

to create the coded symbol interfering with the current received symbol such that

$$m = \begin{cases} 0 \text{ if } & l = 0 \\ 1 \text{ if } & l = 1,2,3 \\ 2 \text{ if } & l = 4,5,6 \\ 3 \text{ if } & l = 7,8,9 \end{cases},$$

$$n = \begin{cases} 0 \text{ if } & l = 0,1 \\ 1 \text{ if } & l = 2,3,4 \\ 2 \text{ if } & l = 5,6,7 \\ 3 \text{ if } & l = 8,9 \end{cases},$$

$$o = \begin{cases} 0 \text{ if } & l = 0,1,2 \\ 1 \text{ if } & l = 3,4,5 \\ 2 \text{ if } & l = 6,7,8 \\ 3 \text{ if } & l = 9 \end{cases}.$$

Finally, since there are always only two equiprobable codewords that can follow each codeword, $P(s_t) = 0.5$. Therefore,

$$\omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) = \frac{1}{2\sqrt{2\pi}\sigma}\left(\frac{-\sum_{i=1}^{k}\Delta_{i,t}}{2\sigma^2}\right), \quad (35)$$

which completely describes the transition from state $S_{j,t-1}$ to $S_{i,t}$.

The model derived for the NI-JED without interleaving can be used to jointly equalize and decode BPSK modulated coded information transmitted through a multipath channel with CIR length L, where the convolutional encoder has constraint length K and the code rate is $R_c = k/n = 1/3$. This model can be extended for higher order modulation alphabets, larger encoder constraint lengths, and different code rates. [f]

5.1.4 Computational complexity

The computational complexity is determined by counting the number of computations performed for each received data block, and expressed in terms of the uncoded block length N_u, the CIR length L and the encoder constraint length K and the modulation alphabet size M. The computational complexity of the NI-JED without interleaving, for the MLSE and MAP algorithms, is determined by

$$CC_{MLSE} = O(8KLN_uM^{Q+(K-1)}) \quad (36)$$

and

$$CC_{MAP} = O(2N_uM^{(Q+(K-1)}(8KL+6)). \quad (37)$$

Figure 6 shows the normalized computational complexity [g] of this joint equalizer and decoder, employing the MLSE algorithm, for a CIR length of $L = 2$ to $L = 10$, constraint lengths $K = 2$, $K = 4$, and $K = 16$, and modulation alphabet sizes of $M = 2$, $M = 4$, $M = 16$, and $M = 64$. Figure 7 shows the complexity of employing the MAP algorithm for the same parameters, where the respective complexity curves are only slightly higher than their corresponding MLSE complexity curves in Figure 6. It is clear that the computational complexity grows exponentially with an increase in channel memory, encoder constraint length, and modulation alphabet size, and it should be clear that computational complexity for long channel memories and large encoder constraint lengths is too high for feasible implementation.

5.2 Block interleaving

In [9], it was demonstrated that equalization and decoding can be performed jointly by transforming the convolution decoder trellis into a super-trellis, and then using this super-trellis to perform equalization while decoding. This is a novel idea in the sense that all the available information is processed as a whole, and there is no exchange of information between independent subunits. This approach achieves optimal performance, because all calculations on the super-trellis are performed with complete information. The only limitation is that the interleaver has to be a block interleaver and that this interleaver has a certain depth limit due to the exponential relationship between the interleaver depth and the computational complexity.

Contrary to this approach, turbo equalization makes use of two independent subunits, namely a MAP equalizer and a MAP decoder, with each unit being supplied with information regarding the decisions made by the other unit [3-5]. This results in assumptions and estimations being made by the respective subunits based on incomplete information, which in turn results in suboptimal calculations during subsequent iterations, ultimately leading to suboptimal performance. Only by iteratively exchanging extrinsic information between the equalizer and decoder, where this information is used as prior information on the calculations in the next iteration, can the performance be increased. However, the turbo equalizer is not limited by the structure of the interleaver, since interleaving and deinterleaving are performed during each turbo iteration.

It was shown in [9] that the NI-JED performs optimally and therefore outperforms the turbo equalizer, but its computational complexity grows unimaginably as the interleaver depth and the CIR length increase. Despite the excellent performance of the NI-JED, it is not a viable solution for practical systems. Although the computational complexity of a conventional turbo equalizer (CTE) is not as high, it is still exponentially related to the encoder constraint length, as well as the channel memory length, but it is not influenced by the structure of the interleaver.

In this section, this author attempts to derive a general model for the NI-JED proposed in [9], which can be presented as a function of encoder constraint

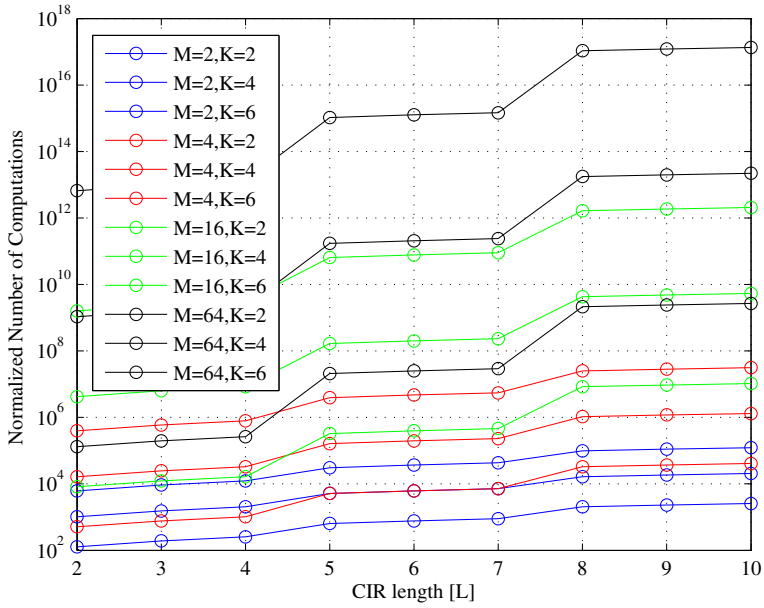

Figure 6 Non-iterative MLSE joint equalizer/decoder normalized computational complexity (without interleaver). Blue circle: $M = 2, K = 2$; Blue square: $M = 2, K = 4$; Blue diamond: $M = 2, K = 6$; Red circle: $M = 4, K = 2$; Red square: $M = 4, K = 4$; Red diamond: $M = 4, K = 6$; Green circle: $M = 16, K = 2$; Green square: $M = 16, K = 4$; Green square: $M = 16, K = 6$; Black diamond: $M = 64, K = 2$; Black circle: $M = 64, K = 4$; Black square: $M = 64, K = 6$.

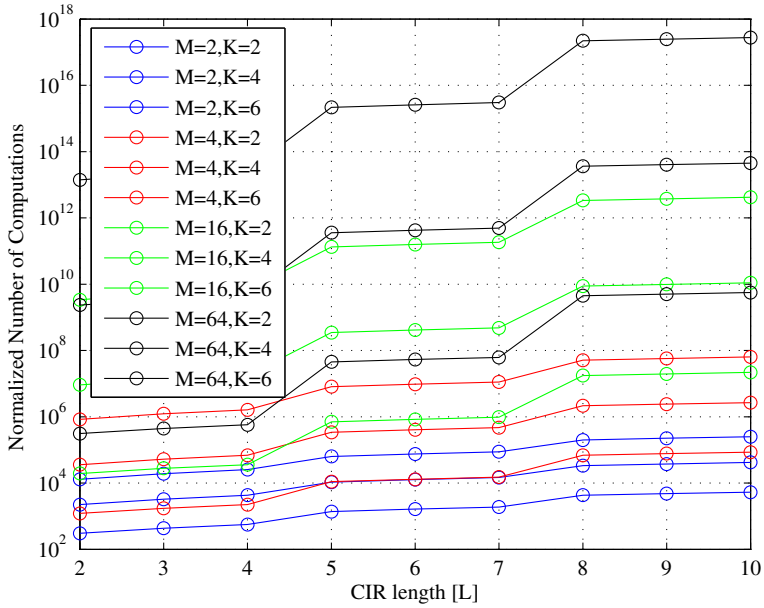

Figure 7 Non-iterative MAP joint equalizer/decoder normalized computational complexity (without interleaver). Blue circle: $M = 2, K = 2$; Blue square: $M = 2, K = 4$; Blue diamond: $M = 2, K = 6$; Red circle: $M = 4, K = 2$; Red square: $M = 4, K = 4$; Red diamond: $M = 4, K = 6$; Green circle: $M = 16, K = 2$; Green square: $M = 16, K = 4$; Green square: $M = 16, K = 6$; Black diamond: $M = 64, K = 2$; Black circle: $M = 64, K = 4$; Black square: $M = 64, K = 6$.

length, interleaver depth, and channel memory length. Knickenberg et al. [9] were very vague as to how the algorithm functions. After explaining the effect of the interleaver on the coded information bits, and the subsequent effect after passing the information through a multipath channel, they simply show that the received symbols contain the uncoded information bits, and state that the MAP algorithm is "invoked". Nothing is said about the fact that the received symbols have to be deinterleaved, nor is it explained how the probabilities between state transitions are determined. The authors of this paper therefore present the NI-JED for various interleaver depths and then derive a general model.

Suppose a communication system encodes a source bit sequence $\mathbf{s}$ of length N_u with a rate $R_c = k/n = 1/3$, constraint length $K = 3$, convolutional encoder with generator polynomials $g_1 = 4\ (100_2)$, $g_2 = 6\ (110_2)$, and $g_3 = 3\ (011_2)$ in order to produce a coded bit sequence $\mathbf{c}$ of length $N_c = N_u/R_c$, which is interleaved using a block interleaver, BPSK modulated, and transmitted through a multipath channel with a CIR $\mathbf{h} = \{h_0, h_1\}$ of length $L = 2$. The encoder produces coded bits $\mathbf{c_t} = \{c_t^{(1)} c_t^{(2)} c_t^{(3)}\}$ from the uncoded bit s_t.

5.2.1 Interleaver depth: $D = n = 3$

When using an $n \times N_u$ interleaver, as shown in Figure 8, the coded sequence

$$\mathbf{c} = \left\{ c_1^{(1)}, c_1^{(2)}, c_1^{(3)}, c_2^{(1)}, c_2^{(2)}, c_2^{(3)}, \ldots, c_{N_u-1}^{(1)}, c_{N_u-1}^{(2)}, c_{N_u-1}^{(3)}, c_{N_u}^{(1)}, c_{N_u}^{(2)}, c_{N_u}^{(3)} \right\},$$

is transformed to

$$\mathbf{c} = \left\{ c_1^{(1)}, c_2^{(1)}, \ldots, c_{t-1}^{(1)}, c_N^{(1)} c_1^{(2)}, c_2^{(2)}, \ldots, c_{t-1}^{(2)}, c_N^{(2)}, c_1^{(3)}, c_2^{(3)}, \ldots, c_{t-1}^{(3)}, c_N^{(3)} \right\}$$

before transmission, grouping all the first, second, and third encoder output bits together after interleaving.

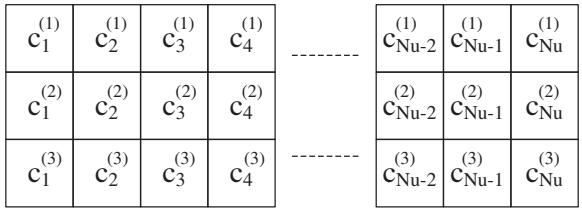

Figure 8 $n \times N_u$ **interleaver.**

During transmission the interleaved coded symbols travel through a multipath channel $\mathbf{h} = \{h_0, h_1\}$ of length $L = 2$ and are received, as shown in the graphical model in Figure 9.[h]

The noiseless received symbols can therefore be expressed in terms of the interleaved coded symbols

$$\begin{aligned} r_t^{(1)} &= c_t^{(1)} h_0 + c_{t-1}^{(1)} h_1 \\ r_t^{(2)} &= c_t^{(2)} h_0 + c_{t-1}^{(2)} h_1, \\ r_t^{(3)} &= c_t^{(3)} h_0 + c_{t-1}^{(3)} h_1 \end{aligned} \tag{38}$$

for $t = 1, 2, \ldots, N_u$, which in turn can be expressed in terms of the uncoded bits

$$\begin{aligned} r_t^{(1)} &= s_t h_0 + s_{t-1} h_1 \\ r_t^{(2)} &= (s_t \oplus s_{t-1}) h_0 + (s_{t-1} \oplus s_{t-2}) h_1 \\ r_t^{(3)} &= (s_{t-1} \oplus s_{t-2}) h_0 + (s_{t-2} \oplus s_{t-3}) h_1 \end{aligned} \quad . \tag{39}$$

From (39) it is clear that $\{s_t, s_{t-1}, s_{t-2}, s_{t-3}\}$ is contained in $\mathbf{r} = \{r_t^{(1)}, r_t^{(2)}, r_t^{(3)}\}$. Therefore, the received symbol sequence $\mathbf{r}$ must be deinterleaved before joint equalization and decoding can commence.[i] Deinterleaving $\mathbf{r}$ will result in the sequence

$$\mathbf{r} = \left\{ r_1^{(1)}, r_2^{(1)}, \ldots, r_{N_u-1}^{(1)}, r_{N_u}^{(1)}, r_1^{(2)}, r_2^{(2)}, \ldots, r_{N_u-1}^{(2)}, r_{N_u}^{(2)}, r_1^{(3)}, r_2^{(3)}, \ldots, r_{N_u-1}^{(3)}, r_{N_u}^{(3)}, \right\}$$

being transformed to

$$\mathbf{r} = \left\{ r_1^{(1)}, r_1^{(2)}, r_1^{(3)}, r_2^{(1)}, r_2^{(2)}, r_2^{(3)}, \ldots, r_{N_u-1}^{(1)}, r_{N_u-1}^{(2)}, r_{N_u-1}^{(3)}, r_{N_u}^{(1)}, r_{N_u}^{(2)}, r_{N_u}^{(3)} \right\},$$

from which it is clear that all $\mathbf{r} = \{r_t^{(1)}, r_t^{(2)}, r_t^{(3)}\}$ are adjacent and can be used to estimate s_t using a trellis-based algorithm, assuming $\{s_{t-1}, s_{t-2}, s_{t-3}\}$ is known from previous states.

The NI-JED uses the sum-product algorithm,[j] as explained in Section 2.2.1, and the number of super-trellis states are determined by the interleaver depth (D), the channel memory length ($L - 1$) as well as the number of encoder memory elements ($K - 1$). Since BPSK modulation is used, the number of states in the super-trellis is $M = 2^{(D/n)(L-1)+(K-1)}$ [9]. Therefore, $M = 8$. The number of states can also be determined by the range of source bits contained in (39) minus one, since the trellis has to model state transitions $\{s_{t-1}, s_{t-2}, s_{t-3}\}$ to $\{s_t, s_{t-1}, s_{t-2}\}$, which when combined yields $\{s_t, s_{t-1}, s_{t-2}, s_{t-3}\}$. Therefore, the number of states can also be determined by $M = 2^{4-1} = 8$.

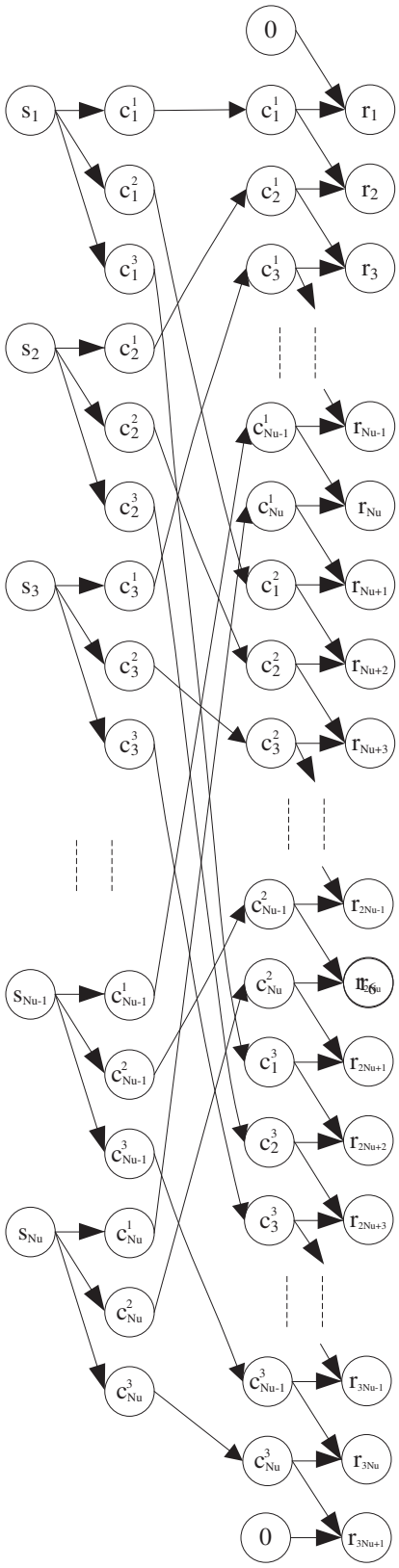

Figure 9 Graphical model for a system with a rate $R_c = 1/3$ convolutional encoder, a $3 \times N_u$ block interleaver and $L = 2$.

The task of the NI-JED is to produced the MAP of each uncoded source symbol at time t from the deinterleaved received symbol sequence, or

$$P(s_t|\mathbf{r}) = \sum_{s_{t^\dagger}:t^\dagger \neq t}^{N_c} P(\mathbf{s}|\mathbf{r}). \tag{40}$$

The probability of a transition from state $S_{j,t-1}$ to state $S_{i,t}$, where $i, j = 1, 2, \ldots, M$ is given by

$$\omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) = P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)}|S_{j,t-1}, S_{i,t})P(s_t), \tag{41}$$

where the symbols associated with states $S_{j,t-1}$ ($\{s_{t-1}, s_{t-2}, s_{t-3}\}$), and $S_{i,t}$ ($\{s_t, s_{t-1}, s_{t-2}\}$) can be combined so that

$$\omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) = P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)}|s_t, s_{t-1}, s_{t-2}, s_{t-3})P(s_t), \tag{42}$$

where

$$\begin{aligned} &P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)}|s_t, s_{t-1}, s_{t-2}, s_{t-3}) \\ &= \frac{1}{\sqrt{2\pi}\sigma}\left(\frac{-\sum_{i=1}^{n}\Delta_{i,t}}{2\sigma^2}\right). \end{aligned} \tag{43}$$

$\Delta_{i,t}$ can be determined by minimizing the receiver equations in (39) such that

$$\begin{aligned} \Delta_{1,t} &= \left|r_t^{(1)} - \textstyle\sum_{l=0}^{L-1} h_l(s_{t-l})\right|^2 \\ \Delta_{2,t} &= \left|r_t^{(2)} - \textstyle\sum_{l=0}^{L-1} h_l(s_{t-l} \oplus s_{(t-1)-l})\right|^2 \\ \Delta_{3,t} &= \left|r_t^{(3)} - \textstyle\sum_{l=0}^{L-1} h_l(s_{(t-1)-l} \oplus s_{(t-2)-l})\right|^2 \end{aligned}, \tag{44}$$

where the calculation of each $\Delta_{i,t}$, $i = 1, 2, \ldots, n$ is determined by the structure of the encoder. Finally, since there are only two equiprobable bits that can be transmitted, $P(s_t) = 0.5$. Therefore,

$$\omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) = \frac{1}{2\sqrt{2\pi}\sigma}\left(\frac{-\sum_{i=1}^{n}\Delta_{i,t}}{2\sigma^2}\right), \tag{45}$$

which completely describes the transition from state $S_{j,t-1}$ to $S_{i,t}$.

5.2.2 Interleaver depth: $D = 2n = 6$

In the above explanation, the depth of the block interleaver was $D = n$. It is subsequently assumed that an interleaver with depth $D = 2n$ is used. Applying the $2n \times N_u/2$ interleaver in Figure 10 to the coded symbols

$$\begin{aligned} \mathbf{c} = \Big\{&c_1^{(1)}, c_1^{(2)}, c_1^{(3)}, c_2^{(1)}, c_2^{(2)}, c_2^{(3)}, \ldots, \\ &c_{N_u-1}^{(1)}, c_{N_u-1}^{(2)}, c_{N_u-1}^{(3)}, c_{N_u}^{(1)}, c_{N_u}^{(2)}, c_{N_u}^{(3)}\Big\}, \end{aligned}$$

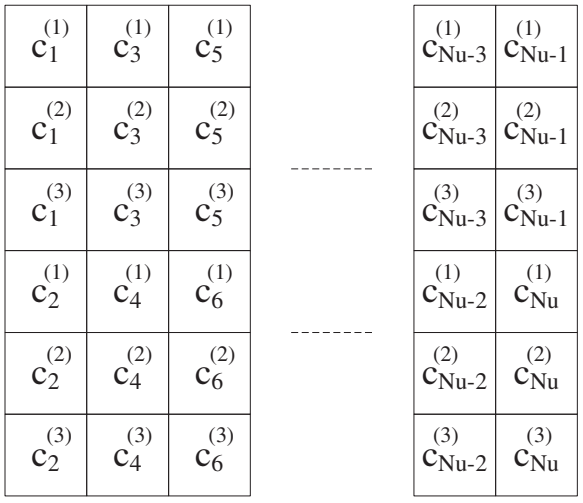

Figure 10 $2n \times N_u/2$ **interleaver.**

transforms it to

$$\begin{aligned}\mathbf{c} = \{&c_1^{(1)}, c_3^{(1)}, \ldots, c_{N_u-1}^{(1)}, c_1^{(2)}, c_3^{(2)}, \ldots, c_{N_c-1}^{(2)}, c_1^{(3)}, c_3^{(3)}, \ldots,\\ &c_{N_c-1}^{(3)}, \ldots, c_2^{(1)}, c_4^{(1)}, \ldots, c_{N_c}^{(1)}, c_2^{(2)}, c_4^{(2)}, \ldots,\\ &c_{N_c}^{(2)}, c_2^{(3)}, c_4^{(3)}, \ldots, c_{N_c}^{(3)}\}.\end{aligned} \tag{46}$$

The received symbols can be expressed in terms of the coded symbols

$$\begin{aligned} r_t^{(1)} &= c_t^{(1)}h_0 + c_{t-2}^{(1)}h_1\\ r_t^{(2)} &= c_t^{(2)}h_0 + c_{t-2}^{(2)}h_1,\\ r_t^{(3)} &= c_t^{(3)}h_0 + c_{t-2}^{(3)}h_1\end{aligned} \tag{47}$$

where $t = 1, 2, \ldots, N_u$, which in turn can be expressed in terms of the uncoded bits

$$\begin{aligned} r_t^{(1)} &= s_t h_0 + s_{t-2}h_1\\ r_t^{(2)} &= (s_t \oplus s_{t-1})h_0 + (s_{t-2} \oplus s_{t-3})h_1\\ r_t^{(3)} &= (s_{t-1} \oplus s_{t-2})h_0 + (s_{t-3} \oplus s_{t-4})h_1\end{aligned} \quad . \tag{48}$$

Comparing (47) and (48) with (38) and (39), respectively, reveals that the system memory has doubled with a twofold increase in interleaver depth (from $D = n$ to $D = 2n$). As before, the received symbol sequence $\mathbf{r}$ must be deinterleaved. Deinterleaving $\mathbf{r}$ will results in the sequence

$$\begin{aligned}\mathbf{r} = \{&r_1^{(1)}, r_3^{(1)}, \ldots, r_{N_u-1}^{(1)}, r_1^{(2)}, r_3^{(2)}, \ldots, r_{N_r-1}^{(2)}, r_1^{(3)}, r_3^{(3)}, \ldots,\\ &r_{N_r-1}^{(3)}, \ldots, r_2^{(1)}, r_4^{(1)}, \ldots, r_{N_r}^{(1)}, r_2^{(2)}, r_4^{(2)}, \ldots,\\ &r_{N_r}^{(2)}, r_2^{(3)}, r_4^{(3)}, \ldots, r_{N_r}^{(3)}\}.\end{aligned}$$

being transformed to

$$\begin{aligned}\mathbf{r} = \{&r_1^{(1)}, r_1^{(2)}, r_1^{(3)}, r_2^{(1)}, r_2^{(2)}, r_2^{(3)}, \ldots\ldots,\\ &r_{N_u-1}^{(1)}, r_{N_u-1}^{(2)}, r_{N_u-1}^{(3)}, r_{N_u}^{(1)}, r_{N_u}^{(2)}, r_{N_u}^{(3)}\},\end{aligned}$$

which ensures that all $\mathbf{r} = \{r_t^{(1)}, r_t^{(2)}, r_t^{(3)}\}$ are adjacent and can be used to estimate s_t, assuming that $\{s_{t-1}, s_{t-2}, s_{t-3}, s_{t-4}\}$ is known from previous states.

As explained earlier, the number of super-trellis states can be determined by the range of source bits in (48) minus one. Counting the range of source bits, minus one, gives the number of states $M = 2^{5-1} = 16$. Alternatively, the number of trellis states can be determined by $M = 2^{D(L-1)+(K-1)}$ which gives $M = 2^{2(1)+2} = 16$. Therefore, for an interleaver with depth $D = 2n$, the calculation of state transition probabilities will require 4 bits per state.

As before, the probability of a transition from state $S_{j,t-1}$ to state $S_{i,t}$, where $i, j = 1, 2, \ldots, M$ is given by

$$\omega_{j\to i,t}(S_{j,t-1}, S_{i,t}) = P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)}|S_{j,t-1}, S_{i,t})P(s_t), \tag{49}$$

which can be written as

$$\begin{aligned}&\omega_{j\to i,t}(S_{j,t-1}, S_{i,t})\\ &\quad = P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)}|s_t, s_{t-1}, s_{t-2}, s_{t-3}, s_{t-4})P(s_t),\end{aligned} \tag{50}$$

where

$$\begin{aligned}&P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)}|s_t, s_{t-1}, s_{t-2}, s_{t-3}, s_{t-4})\\ &=\frac{1}{\sqrt{2\pi}\sigma}\left(\frac{-\sum_{i=1}^{n}\Delta_{i,t}}{2\sigma^2}\right).\end{aligned} \tag{51}$$

$\Delta_{i,t}$ can be determined by minimizing the receiver equations in (48) such that

$$\begin{aligned}\Delta_{1,t} &= \left|r_t^{(1)} - \textstyle\sum_{l=0}^{L-1} h_l(s_{t-2l})\right|^2\\ \Delta_{2,t} &= \left|r_t^{(1)} - \textstyle\sum_{l=0}^{L-1} h_l(s_{t-2l} \oplus s_{(t-1)-2l})\right|^2\\ \Delta_{3,t} &= \left|r_t^{(1)} - \textstyle\sum_{l=0}^{L-1} h_l(s_{(t-1)-2l} \oplus s_{(t-2)-2l})\right|^2\end{aligned} \quad . \tag{52}$$

By comparing (52) and (44), it is already clear that the interleaver depth has an effect on the computational complexity of the NI-JED. Apart from the increase in the number of super-trellis states as a function of interleaver depth, this is the only part of the algorithm that is different from when the interleaver depth is $D = n$. Therefore, the transition probabilities are calculated as before.

5.2.3 Interleaver depth: $D = 3n = 9$

In order to derive a general model for the NI-JED, the system parameters with different interleaver depths are ana-

lyzed. Here it is assumed that the $3n \times N_u/3$ interleaver in Figure 11 is used. Interleaving the coded symbols

$$\mathbf{c} = \{c_1^{(1)}, c_1^{(2)}, c_1^{(3)}, c_2^{(1)}, c_2^{(2)}, c_2^{(3)}, \ldots, c_{N_u-1}^{(1)}, c_{N_u-1}^{(2)}, c_{N_u-1}^{(3)}, c_{N_u}^{(1)}, c_{N_u}^{(2)}, c_{N_u}^{(3)}\},$$

with a $3n \times N_u/3$ interleaver, will result in

$$\begin{aligned}\mathbf{c} = \{\ & c_1^{(1)}, c_4^{(1)}, c_7^{(1)}, \ldots, c_{N_u-2}^{(1)}, c_1^{(2)}, c_4^{(2)}, c_7^{(2)}, \ldots, \\ & c_{N_u-2}^{(2)}, c_1^{(3)}, c_4^{(3)}, c_7^{(3)}, \ldots, c_{N_u-2}^{(3)}, \ldots \\ & c_2^{(1)}, c_5^{(1)}, c_6^{(1)}, \ldots, c_{N_u-1}^{(1)}, c_2^{(2)}, c_5^{(2)}, c_6^{(2)}, \ldots, \\ & c_{N_u-1}^{(2)}, c_2^{(3)}, c_5^{(3)}, c_6^{(3)}, \ldots, c_{N_u-1}^{(3)}, \ldots \\ & c_3^{(1)}, c_6^{(1)}, c_9^{(1)}, \ldots, c_{N_u}^{(1)}, c_2^{(2)}, c_5^{(2)}, c_9^{(2)}, \ldots, \\ & c_{N_u}^{(2)}, c_2^{(3)}, c_5^{(3)}, c_9^{(3)}, \ldots, c_{N_u}^{(3)}\ \}\end{aligned}.$$

Again the received symbols can be expressed in terms of the coded symbols

$$\begin{aligned} r_t^{(1)} &= c_t^{(1)} h_0 + c_{t-3}^{(1)} h_1 \\ r_t^{(2)} &= c_t^{(2)} h_0 + c_{t-3}^{(2)} h_1, \\ r_t^{(3)} &= c_t^{(3)} h_0 + c_{t-3}^{(3)} h_1 \end{aligned} \tag{53}$$

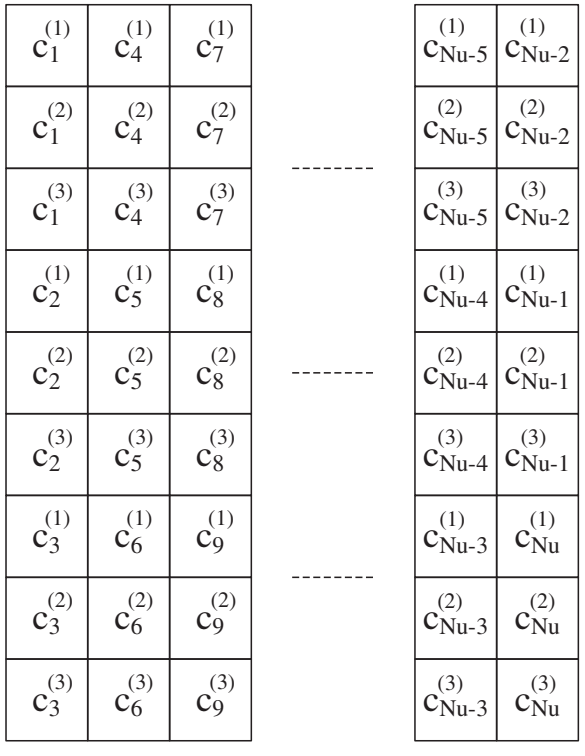

Figure 11 $3n \times N_u/3$ **interleaver.**

where $t = 1, 2, \ldots, N_u$, which can be expressed in terms of the uncoded bits

$$\begin{aligned} r_t^{(1)} &= s_t h_0 + s_{t-3} h_1 \\ r_t^{(2)} &= (s_t \oplus s_{t-1}) h_0 + (s_{t-3} \oplus s_{t-4}) h_1 \\ r_t^{(3)} &= (s_{t-1} \oplus s_{t-2}) h_0 + (s_{t-4} \oplus s_{t-5}) h_1 \end{aligned}, \tag{54}$$

from which it is clear that the system memory is once again affected by the increase in interleaver depth. Deinterleaving $\mathbf{r}$

$$\begin{aligned}\mathbf{r} = \{\ & r_1^{(1)}, r_4^{(1)}, r_7^{(1)}, \ldots, r_{N_u-2}^{(1)}, r_1^{(2)}, r_4^{(2)}, r_7^{(2)}, \ldots, \\ & r_{N_u-2}^{(2)}, r_1^{(3)}, r_4^{(3)}, r_7^{(3)}, \ldots, r_{N_u-2}^{(3)}, \ldots \\ & r_2^{(1)}, r_5^{(1)}, r_6^{(1)}, \ldots, r_{N_u-1}^{(1)}, r_2^{(2)}, r_5^{(2)}, r_6^{(2)}, \ldots, \\ & r_{N_u-1}^{(2)}, r_2^{(3)}, r_5^{(3)}, r_6^{(3)}, \ldots, r_{N_u-1}^{(3)}, \ldots \\ & r_3^{(1)}, r_6^{(1)}, r_9^{(1)}, \ldots, r_{N_u}^{(1)}, r_3^{(2)}, r_6^{(2)}, r_9^{(2)}, \ldots, \\ & r_{N_u}^{(2)}, r_3^{(3)}, r_6^{(3)}, r_9^{(3)}, \ldots, r_{N_u}^{(3)}\ \}\end{aligned}$$

results in

$$\mathbf{r} = \{r_1^{(1)}, r_1^{(2)}, r_1^{(3)}, r_2^{(1)}, r_2^{(2)}, r_2^{(3)}, \ldots, r_{N_u-1}^{(1)}, r_{N_u-1}^{(2)}, r_{N_u-1}^{(3)}, r_{N_u}^{(1)}, r_{N_u}^{(2)}, r_{N_u}^{(3)}\},$$

which ensures that all $\mathbf{r} = \{r_t^{(1)}, r_t^{(2)}, r_t^{(3)}\}$ are adjacent and can be used to estimate s_t, assuming that $\{s_{t-1}, s_{t-2}, s_{t-3}, s_{t-4}, s_{t-5}\}$ is known from previous states. Therefore, the number of super-trellis states for an interleaver depth of $D = 3n$ and channel memory length $L-1$ is $M = 2^{D(L-1)+(K-1)}$ which gives $M = 2^{3(1)+2} = 32$.

The probability of a transition from state $S_{j,t-1}$ to state $S_{i,t}$, where $i, j = 1, 2, \ldots, M$ is given by

$$\omega_{j \to i,t}(S_{j,t-1}, S_{i,t}) = P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)} | S_{j,t-1}, S_{i,t}) P(s_t), \tag{55}$$

which can be written as

$$\begin{aligned} &\omega_{j \to i,t}(S_{j,t-1}, S_{i,t}) \\ &= P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)} | s_t, s_{t-1}, s_{t-2}, s_{t-3}, s_{t-4}, s_{t-5}) P(s_t). \end{aligned} \tag{56}$$

where

$$\begin{aligned} &P(r_t^{(1)}, r_t^{(2)}, r_t^{(3)} | s_t, s_{t-1}, s_{t-2}, s_{t-3}, s_{t-4}, s_{t-5}) \\ &= \frac{1}{\sqrt{2\pi}\sigma} \left(\frac{-\sum_{i=1}^{n} \Delta_{i,t}}{2\sigma^2} \right). \end{aligned} \tag{57}$$

$\Delta_{i,t}$ can be determined by minimizing the receiver equations in (48) such that

$$
\begin{aligned}
l\Delta_{1,t} &= \left| r_t^{(1)} - \sum_{l=0}^{L-1} h_l(s_{t-3l}) \right|^2 \\
\Delta_{2,t} &= \left| r_t^{(1)} - \sum_{l=0}^{L-1} h_l(s_{t-3l} \oplus s_{(t-1)-3l}) \right|^2 \\
\Delta_{3,t} &= \left| r_t^{(1)} - \sum_{l=0}^{L-1} h_l(s_{(t-1)-3l} \oplus s_{(t-2)-3l}) \right|^2
\end{aligned}
\quad . \tag{58}
$$

5.2.4 Interleaver depth: D = dn

After deriving the NI-JED for various interleaver depths it is now possible to present a general model for this algorithm for any $nk \times N_u/d$ interleaver. Even though the channel memory length $L-1$ remained fixed at $L-1=1$, and the number of encoder output bits remained fixed at $n=3$ throughout the analysis, it is easy to generalize the model for any L and n, as will be shown.

When an $nk \times N_u/d$ interleaver is used to interleave the coded symbol sequence

$$
\begin{aligned}
\mathbf{c} = \{ & c_1^{(1)}, c_1^{(2)}, c_1^{(3)}, c_2^{(1)}, c_2^{(2)}, c_2^{(3)}, \ldots\ldots, \\
& c_{N_u-1}^{(1)}, c_{N_u-1}^{(2)}, c_{N_u-1}^{(3)}, c_{N_u}^{(1)}, c_{N_u}^{(2)}, c_{N_u}^{(3)}\},
\end{aligned}
$$

encoded with an encoder that produces n output bits, the result will be

$$
\begin{aligned}
\mathbf{c} = \{ \; & c_1^{(1)}, c_{1+d}^{(1)}, c_{1+2d}^{(1)}, \ldots, c_{N_u-(d-1)}^{(1)}, c_1^{(2)}, c_{1+d}^{(2)}, c_{1+2d}^{(2)}, \ldots, \\
& c_{N_u-(d-1)}^{(2)}, \ldots, c_1^{(n)}, c_{1+d}^{(n)}, c_{1+2d}^{(n)}, \ldots, c_{N_u-(d-1)}^{(n)}, \cdots \\
& c_2^{(1)}, c_{2+d}^{(1)}, c_{1+2d}^{(1)}, \ldots, c_{N_u-(d-2)}^{(1)}, c_2^{(2)}, c_{2+d}^{(2)}, c_{2+2d}^{(2)}, \ldots, \\
& c_{N_u-(d-2)}^{(2)}, \ldots, c_2^{(n)}, c_{2+d}^{(n)}, c_{2+2d}^{(n)}, \ldots, c_{N_u-(d-2)}^{(n)}, \cdots \\
& \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\
& c_d^{(1)}, c_{d+d}^{(1)}, c_{d+2d}^{(1)}, \ldots, c_{N_u}^{(1)}, c_d^{(2)}, c_{d+d}^{(2)}, c_{d+2d}^{(2)}, \ldots, \\
& c_{N_u}^{(2)}, \ldots, c_d^{(n)}, c_{d+d}^{(n)}, c_{d+3d}^{(n)}, \ldots, c_{N_u}^{(n)}\}.
\end{aligned}
$$

Transmitting the interleaved code symbol sequence through a multipath channel with a CIR length of L, where the encoder produces n output bits at a rate of $R_c = 1/n$, the received symbols can be expressed in terms of the coded symbols

$$
\begin{aligned}
r_t^{(1)} &= c_t^{(1)}h_0 + c_{t-d}^{(1)}h_1 + c_{t-2d}^{(1)}h_2 + \cdots + c_{t-(L-1)d}^{(1)}h_{L-1} \\
r_t^{(2)} &= c_t^{(2)}h_0 + c_{t-d}^{(2)}h_1 + c_{t-2d}^{(2)}h_2 + \cdots + c_{t-(L-1)d}^{(2)}h_{L-1} \\
&\vdots \qquad\quad \vdots \qquad\quad \vdots \qquad\quad \vdots \\
r_t^{(n)} &= c_t^{(n)}h_0 + c_{t-d}^{(n)}h_1 + c_{t-2d}^{(n)}h_2 + \cdots + c_{t-(L-1)d}^{(n)}h_{L-1}
\end{aligned}
\quad , \tag{59}
$$

which can be rewritten as

$$
\begin{aligned}
r_t^{(1)} &= \textstyle\sum_{l=0}^{L-1} c_{t-ld}^{(1)}h_l \\
r_t^{(2)} &= \textstyle\sum_{l=0}^{L-1} c_{t-ld}^{(2)}h_l \\
&\vdots \qquad\quad \vdots \\
r_t^{(n)} &= \textstyle\sum_{l=0}^{L-1} c_{t-ld}^{(3)}h_l
\end{aligned}
\quad , \tag{60}
$$

which can be expressed in terms of the uncoded bits

$$
\begin{aligned}
r_t^{(1)} &= \textstyle\sum_{l=0}^{L-1} \big((g_1^{(1)}s_{t-ld}) \oplus (g_1^{(2)}s_{(t-1)-ld}) \\
&\quad \oplus (g_1^{(3)}s_{(t-2)-ld}) \oplus \cdots \oplus (g_1^{(K)}s_{(t-(K-1))-ld}) \big) h_l \\
r_t^{(2)} &= \textstyle\sum_{l=0}^{L-1} \big((g_2^{(1)}s_{t-ld}) \oplus (g_2^{(2)}s_{(t-1)-ld}) \\
&\quad \oplus (g_2^{(3)}s_{(t-2)-ld}) \oplus \cdots \oplus (g_2^{(K)}s_{(t-(K-1))-ld}) \big) h_l , \\
&\vdots \qquad\quad \vdots \qquad\quad \vdots \qquad\quad \vdots \qquad\quad \vdots \qquad\quad \vdots \\
r_t^{(n)} &= \textstyle\sum_{l=0}^{L-1} \big((g_n^{(1)}s_{t-ld}) \oplus (g_n^{(2)}s_{(t-1)-ld}) \\
&\quad \oplus (g_n^{(3)}s_{(t-2)-ld}) \oplus \cdots \oplus (g_n^{(K)}s_{(t-(K-1))-ld}) \big) h_l
\end{aligned}
\tag{61}
$$

where K is the encoder constraint length and generator polynomials $g_1 = \{g_1^{(1)}, g_1^{(2)}, \ldots, g_1^{(K)}\}$, $g_2 = \{g_2^{(1)}, g_2^{(2)}, \ldots, g_2^{(K)}\}$ and $g_n = \{g_n^{(1)}, g_n^{(2)}, \ldots, g_n^{(K)}\}$. Deinterleaving $\mathbf{r}$ will result in the sequence

$$
\begin{aligned}
\mathbf{r} = \{ \; & r_1^{(1)}, r_{1+d}^{(1)}, r_{1+2d}^{(1)}, \ldots, r_{N_u-(d-1)}^{(1)}, r_1^{(2)}, r_{1+d}^{(2)}, r_{1+2d}^{(2)}, \ldots, \\
& r_{N_u-(d-1)}^{(2)}, \ldots, r_1^{(n)}, r_{1+d}^{(n)}, r_{1+2d}^{(n)}, \ldots, r_{N_u-(d-1)}^{(n)}, \cdots \\
& r_2^{(1)}, r_{2+d}^{(1)}, r_{1+2d}^{(1)}, \ldots, r_{N_u-(d-2)}^{(1)}, r_2^{(2)}, r_{2+d}^{(2)}, r_{2+2d}^{(2)}, \ldots, \\
& r_{N_u-(d-2)}^{(2)}, \ldots, r_2^{(n)}, r_{2+d}^{(n)}, r_{2+2d}^{(n)}, \ldots, r_{N_u-(d-2)}^{(n)}, \cdots \\
& \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\
& r_d^{(1)}, r_{d+d}^{(1)}, r_{d+2d}^{(1)}, \ldots, r_{N_u}^{(1)}, r_d^{(2)}, r_{d+d}^{(2)}, r_{d+2d}^{(2)}, \ldots, \\
& r_{N_u}^{(2)}, \ldots, r_d^{(n)}, r_{d+d}^{(n)}, r_{d+3d}^{(n)}, \ldots, r_{N_u}^{(n)}\}.
\end{aligned}
$$

being transformed to

$$
\begin{aligned}
\mathbf{r} = \{ & r_1^{(1)}, \ldots, r_1^{(n)}, r_2^{(1)}, \ldots, r_2^{(n)}, \ldots, \\
& r_{N_u-1}^{(1)}, \ldots, r_{N_u-1}^{(n)}, r_{N_u}^{(1)}, r_{N_u}^{(2)}, r_{N_u}^{(3)}\},
\end{aligned}
$$

from which it is clear that all $\mathbf{r} = \{r_t^{(1)}, \ldots, r_t^{(n)}\}$ are adjacent and can be used to estimate s_t using a trellis-based algorithm, assuming $\{s_{t-1}, s_{t-2}, s_{t-3}, \ldots, s_{t-(D(L-1)+(K-1))}\}$ is known from previous states. The number of super-trellis states for an interleaver depth of $D = dn$, channel memory length $L-1$, and encoder constraint length K, is $M = 2^{D(L-1)+(K-1)}$.

The probability of a transition from state $S_{j,t-1}$ to state $S_{i,t}$, where $i, j = 1, 2, \ldots, M$ is given by

$$
\omega_{j \to i,t}(S_{j,t-1}, S_{i,t}) = P(r_t^{(1)}, \ldots, r_t^{(n)} | S_{j,t-1}, S_{i,t}) P(s_t), \tag{62}
$$

which can be written as

$$\begin{aligned}&\omega_{j\to i,t}(S_{j,t-1}, S_{i,t})\\&= P(r_t^{(1)}, \ldots, r_t^{(n)} | s_t, s_{t-1}, s_{t-2}, \ldots, s_{t-(d(L-1)+(K-1))}) P(s_t),\end{aligned} \tag{63}$$

where

$$\begin{aligned}&P(r_t^{(1)}, \ldots, r_t^{(n)} | s_t, s_{t-1}, s_{t-2}, \ldots, s_{t-(d(L-1)+(K-1))})\\&= \frac{1}{\sqrt{2\pi}\sigma} \left(\frac{-\sum_{i=1}^{n)} \Delta_{i,t}}{2\sigma^2} \right)\end{aligned} \tag{64}$$

and $P(s_t) = 0.5$ as before.

$\Delta_{i,t}$ can be determined by minimizing the receiver equations in (61) such that

$$\begin{aligned}\Delta_{1,t} &= \left| r_t^{(1)} - \sum^{L-1)_{l=0}} \left((g_1^{(1)} s_{t-ld}) \oplus (g_1^{(2)} s_{(t-1)-ld}) \right.\right.\\&\quad \left.\left. \oplus \ldots \oplus (g_1^{(K)} s_{(t-(K-1))-ld}) \right) h_l \right|^2\\ \Delta_{2,t} &= \left| r_t^{(2)} - \sum_{l=0}^{L-1)} \left((g_2^{(1)} s_{t-ld}) \oplus (g_2^{(2)} s_{(t-1)-ld}) \right.\right.\\&\quad \left.\left. \oplus \ldots \oplus (g_2^{(K)} s_{(t-(K-1))-ld}) \right) h_l \right|^2\\ &\vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots\\ \Delta_{n,t} &= \left| r_t^{(n)} - \sum_{l=0}^{L-1)} \left((g_n^{(1)} s_{t-ld}) \oplus (g_n^{(2)} s_{(t-1)-ld}) \right.\right.\\&\quad \left.\left. \oplus \ldots \oplus (g_n^{(K)} s_{(t-(K-1))-ld}) \right) h_l \right|^2\end{aligned} \tag{65}$$

This algorithm can therefore be used to jointly, non-iteratively equalize, and decode BPSK modulated information, encoded with a rate $R_c = 1/n$, constraint length K convolutional encoder, transmitted through a multipath channel with a CIR length of L, using an interleaver with depth $D = dn$. This concludes the derivation of a general model for the NI-JED.

5.2.5 Computational complexity

The computational complexity of the NI-JED, with block interleaving, is determined by counting the number of computations performed for each data block received, and expressed in terms of the uncoded block length N_u, the CIR length L and the encoder constraint length K, the modulation alphabet size M, and the interleaver depth D. The computational complexity of the NI-JED without interleaving, using the MAP algorithm, is determined by

$$CC_{\text{MAP}} = O(2N_u M^{D(L-1)+(K-1)}(8KL + 6)). \tag{66}$$

Figure 12 shows the normalized computational complexity for a CIR length of $L = 2$ to $L = 10$, encoder constraint lengths $K = 2$, $K = 4$, $K = 6$, a modulation alphabet size of $M = 2$ and interleaver depths of $D = 3$, $D = 6$, $D = 9$, and $D = 12$. It is clear that the computational complexity grows exponentially with an increase in channel memory, encoder constraint length and interleaver depth. Figure 13 shows the normalized computational complexity for the same parameters but with a fixed interleaver depth of $D = 6$ and varying modulation alphabet sizes of $M = 2$, $M = 4$, $M = 16$, and $M = 64$, resulting in an increase in complexity with an increase in modulation alphabet size.

6 Simulation results

The performance of the NI-JED is evaluated for BPSK modulation with and without interleaving, and it is compared to that of a CTE, where the number of CTE iterations is $Z = 10$. The CIR $\mathbf{h} = \{h_0, h_1, \ldots, h_{L-1}\}$ of length L is normalized such that $\mathbf{h}'\mathbf{h} = 1$, the uncoded and coded block lengths are $N_u = 600$ and $N_u = 1800$, respectively, the various channel lengths are $L = 2$, $L = 3$, and $L = 4$, and interleaver depths of $D = 1$, $D = k$, $D = 2k$, and $D = 3k$ were used, where $k = 3$ is the number of encoder output bits. Uncoded and coded data block lengths are $N_u = 600$ and $N_c = 1800$, respectively.

Figure 14 shows the BER performance of the NI-JED compared to that of the CTE, for various interleaver depths and a channel length of $L = 2$. From Figure 14 it can be seen that the performance of both algorithms improves with an increase in the interleaver depth, except for an increase from $D = 1$ to $D = 3$. It is clear that the NI-JED outperforms the CTE, even thought the CTE uses multiple iterations, while the NI-JED performs close to the coded additive white Gaussian noise (AWGN) bound when the interleaver depth is $D = 9$. In Figures 15 and 16, the performance of the NI-JED is again compared to that of the CTE for various interleaver depths, for $L = 3$ and $L = 4$, respectively. As before, the NI-JED outperforms the CTE. From Figures reffig14, 15, and 16, it is clear that the NI-JED is superior in terms of performance, and it is in fact optimal. Even though the NI-JED outperforms the CTE, its vast computational complexity inhibits it from finding practical application. The CTE is therefore used as an alternative solutions in practical systems.

7 Conclusions

In this article, optimal equalization and decoding using the MLSE (min-sum) and MAP (sum-product) algorithms were discussed. It was shown how the MLSE algorithm can be used to determine the most likely sequence of estimates, while the MAP algorithm can be used to determine optimal posterior probabilities regarding the transmitted symbols or codewords. NI-JED was also discussed, first assuming no interleaving and then assuming that special block interleavers were used for interleaving. As a result, a general model was derived for systems transmitting convolutionally encoded BPSK modulated information through a multipath channel of length L, where the information is interleaved with an interleaver of depth $D = dk$,

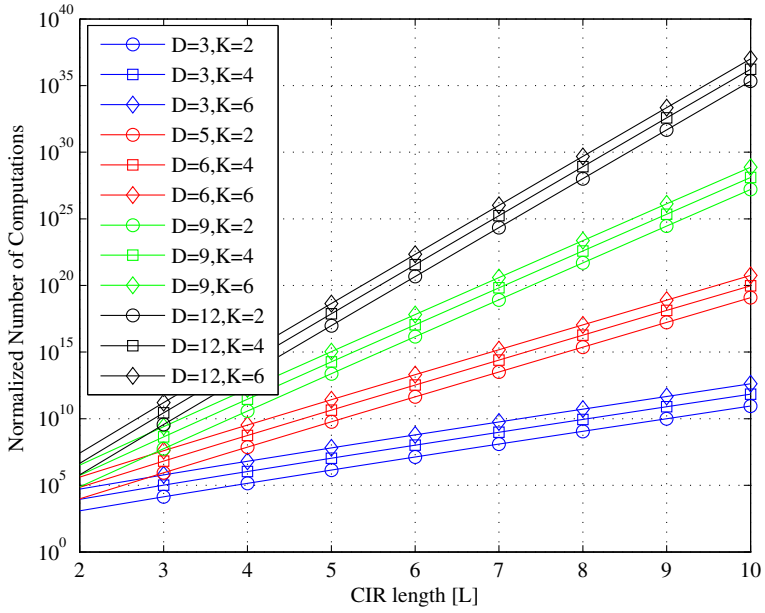

Figure 12 NI-JED normalized computational complexity (with a block interleaver) for $M = 2$. Blue circle: $D = 3, K = 2$; Blue square:$D = 3, K = 4$; Blue diamond: $D = 3, K = 6$; Red circle: $D = 6, K = 2$; Red square: $D = 6, K = 4$; Red diamond: $D = 6, K = 6$; Green circle: $D = 9, K = 2$; Green square: $D = 9, K = 4$; Green square: $D = 9, K = 6$; Black diamond: $D = 12, K = 2$; Black circle: $D = 12, K = 4$; Black square: $D = 12, K = 6$.

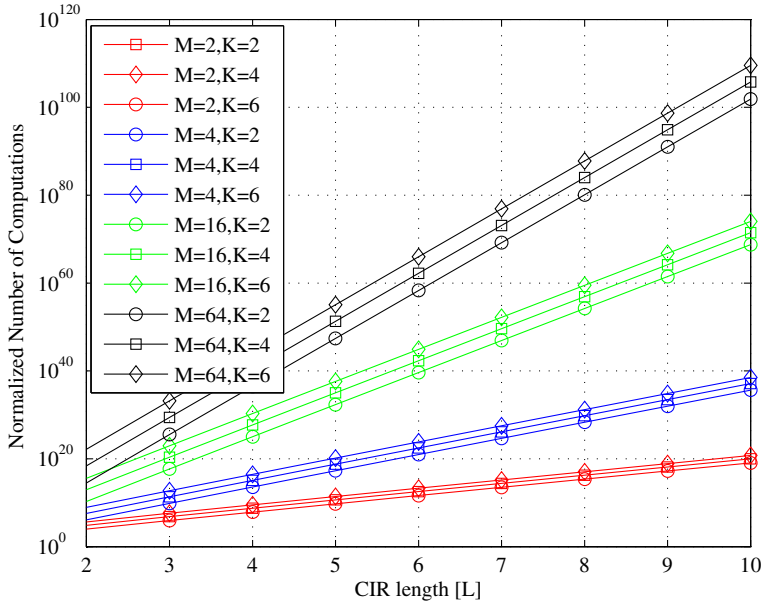

Figure 13 NI-JED normalized computational complexity (with a block interleaver) for $D = 6$. Red circle: $M = 2, K = 2$; Red square: $M = 2, K = 4$; Red diamond: $M = 2, K = 6$; Blue circle: $M = 4, K = 2$; Blue square: $M = 4, K = 4$; Blue diamond: $M = 4, K = 6$; Green circle:$M = 16, K = 2$; Green square: $M = 16, K = 4$; Green square: $M = 16, K = 6$; Black diamond: $M = 64, K = 2$; Black circle: $M = 64, K = 4$; Black square:$M = 64, K = 6$.

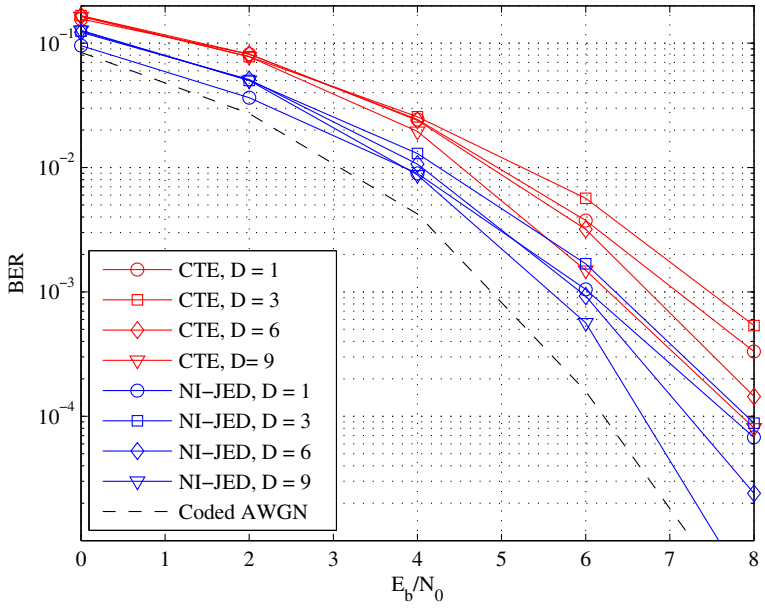

Figure 14 NI-JED and CTE performance for interleaver depths $D = 1$, $D = k$, $D = 2k$ and $D = 3k$, and CIR length $L = 2$. Red circle: CTE, $D = 1$; Red square: CTE, $D = 3$; Red diamond: CTE, $D = 6$; Red triangle: CTE, $D = 9$; Blue circle: NI-JED, $D = 1$; Blue square: NI-JED, $D = 3$; Blue diamond: NI-JED, $D = 6$; Blue triangle: NI-JED, $D = 9$; Black dashed: Coded AWGN.

and where the uncoded information is encoded with a rate $R_c = 1/k$ encoder with constraint length K. The computational complexity was analyzed by counting the number of computations performed, given certain system parameters. The complexity of the NI-JED without interleaving grows exponentially with an increase in either channel memory length or encoder constraint length, while the complexity of the NI-JED with block interleaving is exponentially related to the channel memory length, encoder constraint length, and the interleaver depth.

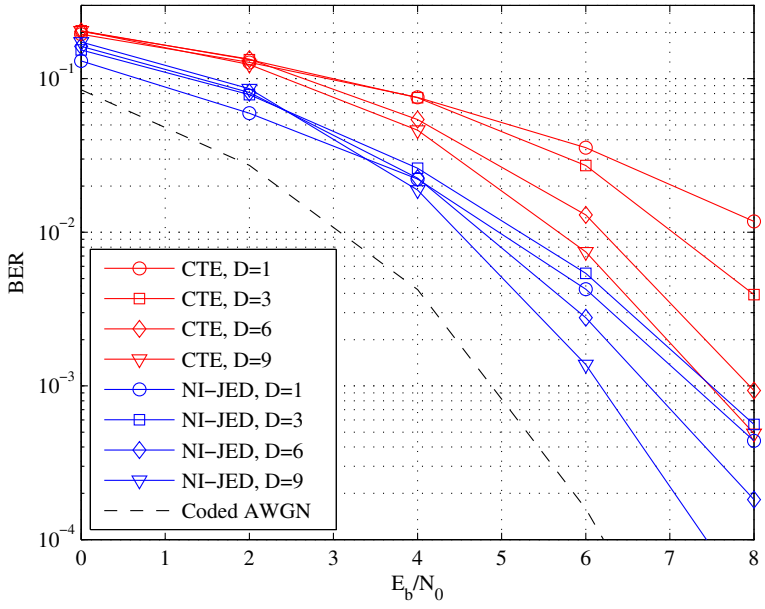

Figure 15 NI-JED and CTE performance for interleaver depths $D = 1$, $D = k$, $D = 2k$ and $D = 3k$, and CIR length $L = 3$. Red circle: CTE, $D = 1$; Red square: CTE, $D = 3$; Red diamond: CTE, $D = 6$; Red triangle: CTE,$D = 9$; Blue circle: NI-JED, $D = 1$; Blue square: NI-JED, $D = 3$; Blue diamond: NI-JED, $D = 6$; Blue triangle: NI-JED, $D = 9$; Black dashed: Coded AWGN.

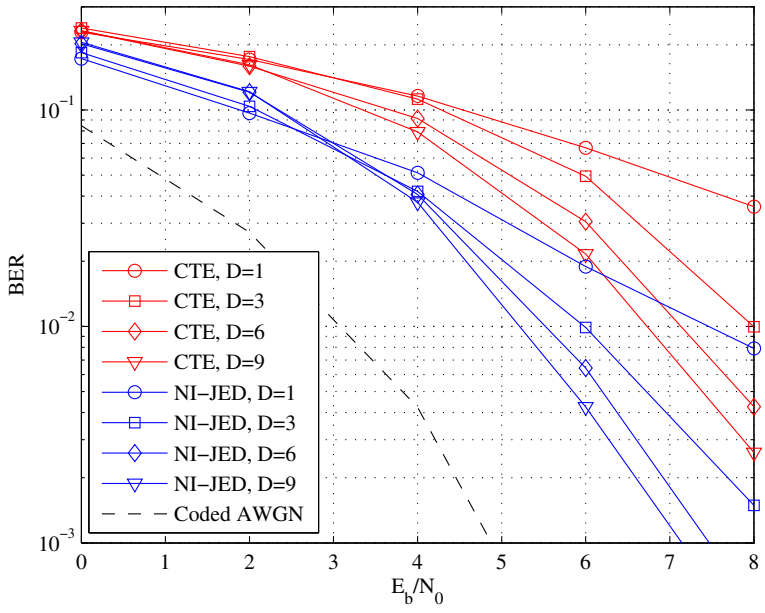

Figure 16 NI-JED and CTE performance for interleaver depths $D = 1$, $D = k$, $D = 2k$ and $D = 3k$, and CIR length $L = 4$. Red circle: CTE, $D = 1$; Red square: CTE, $D = 3$; Red diamond: CTE, $D = 6$; Red triangle: CTE, $D = 9$; Blue circle: NI-JED, $D = 1$; Blue square: NI-JED, $D = 3$; Blue diamond: NI-JED, $D = 6$; Blue triangle: NI-JED, $D = 9$; Black dashed: Coded AWGN.

From these analyses, it is clear that NI-JED is extremely expensive in terms of the number of computations required, even for moderate channel memory lengths, encoder constraint lengths, and interleaver depths. In order to achieve acceptable performance in a multipath fading environment, block interleavers with depths of multiple orders of the encoder output length are required to separate adjacent coded symbols sufficiently. Under these conditions the NI-JED becomes infeasible. Ideally, a random interleaver will allow for maximum performance gains, but the NI-JED cannot be applied when interleaving is performed with a random interleaver. The CTE is therefore applied in systems transmitting randomly interleaved coded information through a multipath channel.

Turbo equalization is used as an alternative to optimal NI-JED, which is not feasible because of computational complexity constraints discussed before, to the extent that approximate inference via the iterative exchange of information is the last resort. Turbo equalization is not optimal, as demonstrated in this article, but it is the best alternative amongst all iterative joint equalization and decoding solutions, as its constituent parts—the MAP equalizer and the MAP decoder—produce optimal posterior estimates about the respective coded and uncoded transmitted symbols.

Endnotes

[a]It is assumed that $GF(2)$ decoding is used as usual.
[b]$d_\xi \in \{-1, 1\}$ for BPSK.
[c]The magnitude of $L(\tilde{d}_t)$ gives an indication of the confidence of that estimate.
[d]K is also known as the constraint length.
[e]During decoding, the output bits $\mathbf{c}_t = \{c_t^{(1)}, \ldots, c_t^{(n)}\}$ are made bipolar because the elements of $\mathbf{c}_t$ are compared to received *symbols*, and not bits.
[f]Joint equalization and decoding using a higher order modulation alphabet will require convolutional encoding to be performed in GF(M), where M is the modulation alphabet size.
[g]The computational complexity is normalized by the number of coded transmitted symbols N_c.
[h]Note that the channel coefficients are not shown in Figure 9.
[i]This step in not mentioned, nor implied in [9], but has been inferred by the author of this article.
[j]The min-sum algorithm can also be used.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Department of Electrical, Electronic and Computer Engineering, University of Pretoria, Pretoria 0002, South Africa. [2]School of Engineering, University of Tasmania, Hobart, TAS 7001, Australia.

Received: 7 February 2013 Accepted: 19 March 2013
Published: 15 April 2013

References

1. L Bahl, J Cocke, F Jelinek, J Raviv, Optimal decoding of linear codes for minimizing symbol error rate. IEEE Trans. Inf. Theory. **IT-20**, 284–287 (1974)

2. J Proakis, *Digital Communications*, 4th edn. (McGraw-Hill, International Edition, New York, 2001)
3. G Bauch, H Khorram, J Hagenauer, in *Proceedings of European Personal Mobile Communications Conference (EPMCC)*. Iterative equalization and decoding in mobile communication systems, (1997), pp. 307–312
4. C Douillard, M Jezequel, C Berrou, A Picart, P Didier, A Glavieux, Iterative correction of intersymbol intereferenceturbo-equalization. Eur. Trans. Telecommun. **6**, 507–511 (1995)
5. R Koetter, M T*ü*chler, A Singer, Turbo equalization. IEEE Signal Process. Mag. **21**, 67–80 (2004)
6. A Viterbi, Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Trans. Inf. Theory. **IT-13**, 260–269 (1967)
7. G Forney, The viterbi algorithm. Proc. IEEE. **61**(3), 268–278 (1973)
8. J Hagenauer, in *Proceedings of the IEEE Global Telecommunications Conference*, vol. 3. A viterbi algorithm with soft-decision outputs and its applications, (1989), pp. 1680–1686
9. A Knickenberg, B Yeap, J Hamorsky, M Breiling, L Hanzo, in *Globecom 1999*, vol. 9. Non-iterative joint channel equalization and channel decoding, (1999), pp. 442–446
10. M Breiling, L Hanzo, The super-trellis structure of turbo codes. IEEE Trans. Inf. Theory. **46**(6), 2212–2228 (2000)
11. G Forney, Maximum likelihood sequence estimation of digital sequences in the presence of intersymbol interference. IEEE Trans. Inf. Theory. **IT-18**, 363–378 (1972)
12. J Pearl, Fusion, propagation, and structuring in belief networks. Elsevier Artif. Intell. **29**(3), 241–288 (1986)
13. D Mackay, *Information Theory, Inference, and Learning Algorithms*. (Cambridge University, Press, Cambridge, MA,2003)
14. Y Zheng, C Xiao, Improved models for the generation of multiple uncorrelated Rayleigh fading waveforms. IEEE Commun. Lett. **6**, 256–258 (2002)

doi:10.1186/1687-6180-2013-79
Cite this article as: Myburgh and Olivier: **A primer on equalization, decoding and non-iterative joint equalization and decoding.** *EURASIP Journal on Advances in Signal Processing* 2013 **2013**:79.