Illuminating the Structure of Code and Decoder of Parallel Concatenated Recursive Systematic (Turbo) Codes

Patrick Robertson *
Institute for Communications Technology
German Aerospace Research Establishment (DLR)
P.O. Box 1116, D-82230 Wessling, Germany.
Tel.: ++49 8153 282808; FAX: ++49 8153 281442
e-mail: patrick@holmes.nt.op.dlr.de

Abstract

Recently, a coding scheme (turbo-codes) was proposed, that achieves almost reliable data communication at signal-to-noise ratios very close to the Shannon-limit. In this paper we show that the associated iterative decoder can be formulated in a simpler fashion by passing information from one decoder to the next using *log-likelihood ratios* as opposed to channel values that need to be normalized. Also no heuristically determined correction parameters are necessary for stable decoding. In addition, we can reduce the average number of iterations needed for the same BER performance by determining when further iterations achieve no more benefit. Furthermore, it seems that the trellis-termination problem appears non-trivial and we give a pragmatic suboptimal solution. We investigate different block sizes and also a hybrid scheme that performs extremely well with less computations. A drawback of the codes has been discovered: the BER curves show a flattening at higher signal-to-noise ratios, this is due to the small minimum distance of the whole code. By analyzing the interleaver used in the encoder we can calculate approximations to the BER at high SNRs. Finally, by careful interleaver manipulation the minimum distance of the code can be increased and the error-coefficient for the remaining small distance events can be further reduced. Furthermore, we have investigated the influence of the interleaver length on the SNR needed to achieve a certain BER. Simulations confirm both the analytical approximation to the BER as well as the method for interleaver design which yields a marked improvement at higher SNR.

1 Introduction

Multidimensional array codes and the associated decoding techniques have received attention recently, as they are promising candidates for communications over very noisy channels. To mention are product codes constructed with simple parity check codes [1], Hamming codes or convolutional codes [2].

*This work was partly carried out at the Communications Research Center, Ottawa, Canada during an exchange program financed in part by GKSS/IB project KAN INF 20.

We will try to shed some light on the basic understanding of the decoder of the recently proposed 'turbo-codes' -a parallel concatenation of recursive systematic punctured convolutional codes with random interleaving between the encoders, and an associated iterative decoding scheme [3]. The proposed scheme is claimed, under certain circumstances, to achieve near Shannon-limit error correction capabilities [3]. However the interface between the output of one decoder, and the input of the next decoder in the iterative decoding scheme was not described sufficiently. Furthermore, the non-trivial problem of trellis termination -important for small block sizes- had not been explicitly mentioned nor a solution given; we shall see that this causes no severe problems even for small block sizes.

We will not repeat the derivation of the modified Bahl *et al.* algorithm (MAP) [4] [3], but will only briefly state the results, in particular the modification to be made to the branch transition probability in order to allow one decoder to make use of the output of the other decoder. By optionally estimating the variance of the decoder output we are able to reduce the average number of iterations needed for the same BER performance by determining when further iterations achieve no more benefit. Furthermore, we will look at simulation results for a range of block sizes. To conclude, we shall present a method for estimating a bound on the BER at high SNR and use it as an optimization criterion for interleaver design.

2 The Encoder and Channel Model

The encoder contains of two recursive systematic binary convolutional encoders (henceforth denoted as component encoders) [3]. In the following let the component encoders be identical (extensions to different component codes are trivial), they have M memory elements. The first encoder operates directly on the data to be transmitted, let this be the sequence $\vec{d} = (d_1, ..., d_N)$ of N bits. The first component encoder has two outputs, one is the sequence of information bits $\vec{x}^{1s} = (x_1^{1s}, ..., x_N^{1s}) = \vec{d}$, since the encoder is systematic. The other is the 'parity information' sequence $\vec{x}^{1p} = (x_1^{1p}, ..., x_N^{1p})$, with $x_k^{1p} = \sum_{i=1}^{M} g_i^{1f} a_{k-i}^1$, where

0-7803-1820-X/94 $4.00 © 1994 IEEE

$(g_1^{1f}, ..., g_M^{1f})$ is the feed-forward generator of the first encoder and $a_k^1 = d_k + \sum_{i=1}^{M} g_i^{1b} a_{k-i}^1$; similarly, $(g_1^{1b}, ..., g_M^{1b})$ is the feed-back generator of the first encoder.

The data input sequence to the second component encoder is now interleaved. Let it be represented by the sequence $\vec{d^I} = (d_1^I, ..., d_N^I)$, with $d_k^I = d_{I_k}$, where $\vec{I} = (I_1, I_2, ..., I_N)$ specifies the interleaver that reorders the information bits feeding the second component encoder. Similarly, the associated de-interleaver is specified by $\vec{I'}$ and $d_k = d_{I'_k}^I$. The systematic sequence $\vec{x}^{2s}$ is not transmitted, only the parity information sequence $\vec{x}^{2p}$, generated in an identical fashion to the first component encoder. Both encoders are assumed to start in the all zero state, i.e. all a_i^1 and a_i^2 are zero initially.

To achieve a higher global rate than $R = 1/3$, the outputs $\vec{x}^{1p}$ and $\vec{x}^{2p}$ are punctured according to a puncturing matrix $\mathbf{P}$. To achieve $R = 1/2$, $\mathbf{P}$ is chosen as $\left[\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right]$. The first row determines when x_k^{1p} is sent and the bottom row when x_k^{2p} is sent. The channel investigated here is the AWGN channel, where after modulation each transmitted symbol is added to an independent noise sample with zero mean and variance $N_0/2$. The receiver inserts those symbols that were punctured with the value zero into the sequence $\vec{R}$. The k-th element of the sequence $\vec{R}$ is defined as

$$R_k = \left(y_k^{1s}, y_k^{1p}, y_k^{2p}\right) = \left(2 \cdot x_k^{1s} - 1 + n_k^{1s}, 2 \cdot x_k^{1p} - 1 + n_k^{1p}, 2 \cdot x_k^{2p} - 1 + n_k^{2p}\right). \quad (1)$$

2.1 Trellis Termination

We have mentioned in the introduction, that a solution to the problem of terminating the trellises of both component encoders has not been stated in [3]. Since the encoders are recursive, it does not suffice to set the last M information bits to zero in order to drive the encoder to the all zero state. The termination sequence necessary, is a function of which state the encoder is in at the point when termination is to begin. If we force the upper encoder to the all zero state, then we can still evaluate the sequence $\vec{d^I}$. However, if we now wish to terminate the second encoder also, this would mean changing the information sequence $\vec{d}$ due to the interleaving between the component encoders. This would affect the state of the first encoder before termination begins, requiring a different termination sequence for the first encoder, resulting in a different $\vec{d^I}$ and so forth. There appears no simple solution to terminate both encoders at the same time using only M bits. In our investigations so far, we terminated the first encoder only, and left the second trellis 'open'. This meant a slight modification to the way the decoder associated with the second encoder is initialized. Due to terminating the upper encoder, we only actually transmit $N - M$ information bits.

3 The Decoder

3.1 The MAP Algorithm

We will not repeat the derivation of the MAP algorithm here, but only state the deviations necessary for a more structured approach to pass information between the decoders. The notation applies to the first decoder in the concatenated scheme, the second decoder can be treated is the same way. In [3], an additional variable Z_k was introduced which was an independent observation of x_k^{1s} that was made available by the other decoder. The first decoder had *three* channel inputs, the second decoder had *two* (it did not use the systematic information directly). It was stated in the paper that normalization is necessary to convert the LLR (the MAP output) to a form suitable for use as an input to the MAP. In the following we will present the MAP algorithm in such as way that estimation of variances is not necessary -we simply convert the LLR output by one decoder to a-priori probabilities to be used by the other decoder.

Let the state of the encoder at time k be S_k, it can take on values between 0 and $2^M - 1$. The bit d_k is associated with the transition from step $k-1$ to step k. Goal of the MAP algorithm is to provide us with the logarithm of the ratio of the a posteriori probability (APP) of each information bit d_k being 1 to the APP of it being 0. In a derivation similar to [3] we obtain:

$$\Lambda(d_k) = \log \frac{\sum_m \sum_{m'} \gamma_1(y_k, m', m) \cdot \alpha_{k-1}(m') \cdot \beta_k(m)}{\sum_m \sum_{m'} \gamma_0(y_k, m', m) \cdot \alpha_{k-1}(m') \cdot \beta_k(m)}. \quad (2)$$

The forward recursion of the MAP can be expressed in a simpler fashion (no splitting of α), as in [4]:

$$\alpha_k(m) = \frac{\sum_{m'} \sum_{i=0}^{1} \gamma_i(y_k, m', m) \cdot \alpha_{k-1}(m')}{\sum_m \sum_{m'} \sum_{i=0}^{1} \gamma_i(y_k, m', m) \cdot \alpha_{k-1}(m')}. \quad (3)$$

A similar equation is obtained for the backward recursion $\beta_k(m)$. The branch transition probabilities are given by

$$\gamma_i(y_k^{1s}, y_k^{1p}, m', m) = p(y_k^{1s} | d_k = i, S_k = m, S_{k-1} = m') \cdot p(y_k^{1p} | d_k = i, S_k = m, S_{k-1} = m') \cdot q(d_k = i | S_k = m, S_{k-1} = m') \cdot Pr\{S_k = m | S_{k-1} = m'\}; \quad (4)$$

$q(d_k = i | S_k = m, S_{k-1} = m')$ is either zero or one depending on whether bit i is associated with the transition from state m' to m. It is in the last component that we use the information of the previous decoder: the probability $Pr\{S_k = m | S_{k-1} = m'\}$ depends directly on the a-priori probability of the information bit d_k [5]. We use the a-priori probability of bit d_k given to us by the previous decoder in

$$Pr\{S_k = m | S_{k-1} = m'\} = \frac{e^{L(d_k)}}{1 + e^{L(d_k)}}, \quad (5)$$

if $q(d_k = 1 | S_k = m, S_{k-1} = m') = 1$; and

$$Pr\{S_k = m | S_{k-1} = m'\} = 1 - \frac{e^{L(d_k)}}{1 + e^{L(d_k)}}, \quad (6)$$

if $q(d_k = 0|S_k = m, S_{k-1} = m') = 1$; otherwise we can assign $Pr\{S_k = m|S_{k-1} = m'\}$ arbitrarily. The term $L(d_k) = L_{2e}(d_k) = L_{2e}(d^I_{I'_k})$ is the *extrinsic* component of the LLR that the other decoder has output for the information bit $d^I_{I'_k} = d_k$. It is used as a-priori information in the current decoder. The other probabilities in (4) are calculated according to

$$p(y_k^{1s}|d_k = i, S_k = m, S_{k-1} = m') = k_s e^{\frac{-(y_k^{1s} - b^s(i,m',m))^2}{N_0}}, \quad (7)$$

$$p(y_k^{1p}|d_k = i, S_k = m, S_{k-1} = m') = k_p e^{\frac{-(y_k^{1p} - b^p(i,m',m))^2}{N_0}}, \quad (8)$$

where k_s and k_p are constants that can be set to 1 for the actual calculations, and $b^{s|p}(i, m', m)$ is the modulator output $2 \cdot x_k^{1s} - 1$ resp. $2 \cdot x_k^{1p} - 1$ associated with the branch from state m' at step $k-1$ to state m at step k if the corresponding d_k was equal to i.

In [3] the concept of *extrinsic* and *intrinsic* information was introduced in order to prevent say, the first decoder from passing on information to the second decoder which had been generated by the second decoder in the first place. In other words, we must ensure that the 'a-priori' information is independent of the other information (observations) being used in the decoder. We can write the MAP output for bit d_k as:

$$\Lambda(d_k) = \log \frac{\sum_m \sum_{m'} \gamma'_1(y_k^{1p}, m', m) \cdot \alpha_{k-1}(m') \cdot \beta_k(m)}{\sum_m \sum_{m'} \gamma'_0(y_k^{1p}, m', m) \cdot \alpha_{k-1}(m') \cdot \beta_k(m)} + L(d_k) + \log \frac{p(y_k^{1s}|d_k = 1)}{p(y_k^{1s}|d_k = 0)}, \quad (9)$$

with $\gamma'_i(y_k^{1p}, m', m) = p(y_k^{1p}|d_k = i, S_k = m, S_{k-1} = m') \cdot q(d_k = i|S_k = m, S_{k-1} = m')$. The second component in (9) is the a-priori term, generated by the previous decoder, and the last components is the systematic term. The first component is the extrinsic component $L_{1e}(d_k)$, and is independent of the a-priori and systematic information *for the bit* d_k.

3.1.1 Initialization

α_0 and β_N must be initialized as follows:

$$\alpha_0(0) = 1, \alpha_0(m \neq 0) = 0; \beta_N(0) = 1, \beta_N(m \neq 0) = 0. \quad (10)$$

Except in the case for the decoder associated with the second encoder, where the trellis is simply left 'open':

$$\alpha_0(0) = 1, \alpha_0(m \neq 0) = 0; \beta_N(m) = \alpha_N(m). \quad (11)$$

In the second case the backward recursion uses the value of the state probabilities generated by the last forward recursion step [6].

3.2 Reducing Computational Requirements by Observing the MAP Output Second Moment

Earlier we have said that one method of making the interface between the two MAP decoders is to use channel values along with the variance of the meta-channel in the evaluation of the branch transition probability. This is unnecessary since we can use the LLR directly. But we can still calculate the signal-to-noise ratio of the meta-channel (at the output of each MAP decoder) in order to decide whether or not we should carry on iterating. The hope is that the variance of the meta-channel should be low, if and only if further iterations are without effect on the number of bit errors in that block.

To do this, let us first remind ourselves what connects channel values and LLRs. Assume a channel model where $x \epsilon \{-1, +1\}$ is transmitted across an AWGN channel with noise variance σ^2, the received value is y. It is easy to show that if ± 1 are equally likely that

$$\Lambda(x) = \log \frac{p(y|x = +1)}{p(y|x = -1)} = \frac{2}{\sigma^2} \cdot y = \log \frac{Pr\{x = +1|y\}}{Pr\{x = -1|y\}}, \quad (12)$$

where $\Lambda(x)$ is the conditional LLR of x given observation y. So the point to note is that there exists a linear relation between the LLR $\Lambda(x)$ and the associated channel observation y: $y = \frac{\sigma^2}{2} \cdot \Lambda(x)$.

We want to evaluate the value of σ.

The PDF of x is $p_{\mathbf{X}}(x) = \frac{1}{2} \cdot \delta(x-1) + \frac{1}{2} \cdot \delta(x+1)$, assuming equally likely ± 1's; the PDF of the noise is $p(n) = c_1 \cdot e^{\frac{-n^2}{2\sigma^2}}$, so the PDF of $\Lambda(x)$ becomes

$$p_{\mathbf{\Lambda}}(\Lambda(x)) = \frac{c_2}{2} \cdot e^{\frac{-(\Lambda(x) - \frac{2}{\sigma^2})^2}{2(\sigma \frac{2}{\sigma^2})^2}} + \frac{c_2}{2} \cdot e^{\frac{-(\Lambda(x) + \frac{2}{\sigma^2})^2}{2(\sigma \frac{2}{\sigma^2})^2}}. \quad (13)$$

Let m_{2L} represent the second moment of the RV $\mathbf{\Lambda(x)}$, it can easily be shown that $m_{2\Lambda} = \frac{1}{2} \cdot \left[\frac{4}{\sigma^2} + (\frac{2}{\sigma^2})^2 + \frac{4}{\sigma^2} + (\frac{-2}{\sigma^2})^2\right]$. Rearranging, we arrive at a quadratic equation for σ^2: $\sigma_2^4 m_{2\Lambda} = 4\sigma^2 + 4$, with the single solution $\sigma^2 = \frac{2+2\sqrt{1+m_{2\Lambda}}}{m_{2\Lambda}}$. So we must estimate the second moment of $\mathbf{\Lambda(x)}$ at every decoding stage, in order to evaluate an estimate of σ^2.

We now have an estimate of the overall quality of the decoded bits for that iteration. For example, a variance of 0.055 is equivalent to an AWGN channel with an SNR of 9.6 dB and should have an average BER of 10^{-5}. The number of bit errors found in a block at each iteration was found to be closely linked to the predicted average value. In our simulations, we have set the threshold to stop iterations to an equivalent channel SNR of either 10 dB or 12.5 dB depending on the BER range we are operating in. In practice we have observed the following: blocks that were hard to decode (i.e. needed many iterations) have a high variance associated with each iteration until they are error-free. No situation has been encountered so far where the variance threshold was reached, yet where more iterations could still have improved performance.

3.3 The Decoder Structure

We are now able to describe how the individual components of the decoder are linked together. Fig. 1 shows the involved variables. The two central units are the (symmetrical) MAP

decoders associated with the two encoders, the extrinsic information passed between them must be interleaved or de-interleaved accordingly. Decoder delays are not taken into account in the figure.

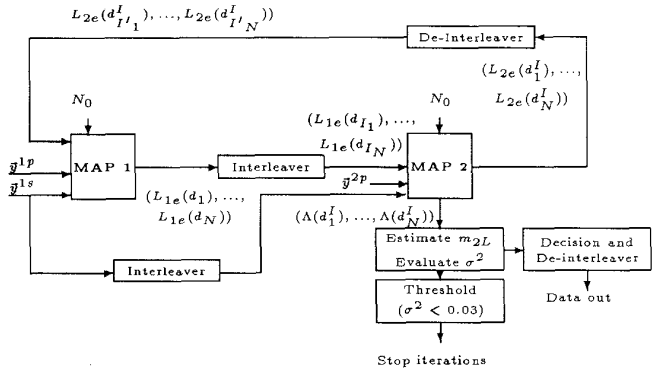

Figure 1: Structure of the Turbo-Decoder.

4 Simulation Results

In order to produce more reliable simulation results in a relatively short time, we have chosen to restrict our main investigations to block sizes $N \leq 10000$ for memory 4 codes. In Fig. 4 we see the performance in terms of BER for $N = 10000$ and $M1 = M2 = 4$, $R = 1/2$, generators 37 and 21 as given in [3]. A random interleaver was used. The parameter is the maximum number of iterations allowed (not the average, which is shown additionally). Note the sharp drop at around 0.75 dB -the BER drops by two orders of magnitude in only 0.1 dB at 18 iterations. Interesting is the drop in the steepness of the curve at 0.9 dB, this is an effect we will also see for smaller block sizes. The dotted line shows the evaluation of the approximate bound (14) introduced in the next section. Never up to the present time have error events been observed that lie above the raw bit error rate of the channel, (for block sizes $N \geq 1000$). This appears to be due to the fact that the decoder makes use of channel inputs at *each* decoding step. If the outputs of the MAP decoders become unreliable, then the decoder will rely more heavily on the channel values. Finally, we see that the average number of iterations needed is smaller for larger SNR.

Since it was observed that a less powerful memory two codes performed better in the first iteration, we tried a hybrid construction where $M = 2$ for the first code and $M = 4$ for the second code. For $N = 10000$ and 18 iterations, we found that this combination is only 0.1 dB worse (at a BER of 10^{-4}) than the original $M = 4$ construction, at a lower overall decoder complexity.

In Fig. 5 we see the BER performance for $N = 100$, 400 and 1024; again $R = 1/2$. Also shown for $N = 100$ and 1024 is the degradation due to not terminating the second trellis (done by transmitting the all zero sequence -both coders are always in the all zero state so we can use (10) to initialize the second decoder), it is more pronounced for the smaller block size. Observe the reduction in the steepness of the curves, particularly for $N = 100$, compared to the large block size presented in [3].

Fig. 6 shows the SNR needed to achieve a BER of 10^{-3}, 10^{-4} and 10^{-5} as a function of the block size N for memory 2 and 4 component codes, for $R = 1/2$. The number of iterations was the same for memory 2 and 4: 18 for $N \geq 4000$; 10 for $1024 \leq N \leq 2000$ and 6 for $N \leq 400$. From these curves we notice the following:

- The steepness of the BER curves increases dramatically with increasing block size, for both memory 2 and 4 component codes.
- At BER 10^{-3} the penalty for using memory 2 is only about 0.25 dB; small when seen in the light of a complexity reduction by a factor of four.
- The loss from using memory two codes becomes more pronounced for block sizes ranging from approximately 400 to 4000 and only if a BER of 10^{-5} is required.
- A saturation seems to have been reached for very large N. For very short blocks the turbo-code construction does not seem worthwhile.

5 Minimum Distance of Whole Code and Interleaver Choice

Let us assume that we are transmitting the all zero data sequence except that two bits are ones (bearing in mind that the code is linear). For a random interleaver, there is a chance that the interleaver maps these two non-zero bits from $\vec{d}$ to $\vec{d}^I$ in such a way that the resulting distance from the all zero code word is small. This is because the recursive codes can be driven out of the all zero state and back to it with a minimum of two ones that are suitably spaced. The above described event corresponds to this being the case for *both* encoders, due to an 'unlucky' mapping of the interleaver shown in Fig. 2. The rate 2/3 memory 4 component code with octal generators 37 and 21, has an error event where the component code word corresponding to the information sequence ...00100001000... is Hamming distance 4 away from the all zero component code word. If the mapping described above really occurs, then this will result in two 'turbo'-code words (the all zero code word and the code word corresponding to $\vec{d}$ = ...00100001000...) to be $4 + 2 = 6$ apart, assuming the worst case puncturing of $\vec{x}^{1p}$ and $\vec{x}^{2p}$ and bearing in mind that the systematic information is only sent once. This is just one example of a set of two code words that are a small distance apart, there are many different such mappings, analysis of which is quite involved. If one knows the number, $a(w, d)$, of such code words of information bit weight w, that are Hamming distance d away from the all zero code word then one can apply the union bound [7] for the bit error rate:

$$P_b \leq \sum_{w=1}^{N} \sum_{d=d_{min}}^{\infty} a(w,d) \cdot \frac{w}{N} \cdot \frac{1}{2} \cdot \mathrm{erfc}\left(\sqrt{d \cdot \frac{R \cdot E_b}{N_0}}\right). \quad (14)$$

The problem lies in finding the distance spectrum $a(w,d)$. For distances d less than 50, and information bit weights w not more than about 6 this can be done by computer search that takes the puncturing, and of course the component code, into account.

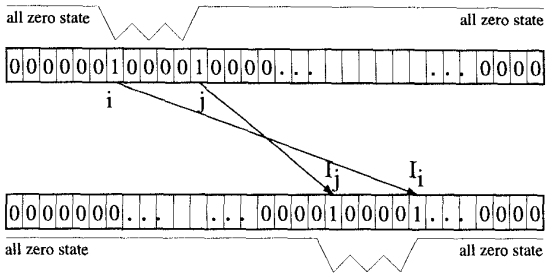

Figure 2: Illustration of a mapping leading to a small distance event. Shown are the sequences $\vec{d}$ (top) and $\vec{d}'$. Also the corresponding paths through the trellises of the two component encoders.

It has been observed that the event leading to the minimum distance of 6 is quite rare, it typically occurs just a few times for any particular interleaver. And assuming that it does occur, the information sequence weight difference between the competing code words is also small. The results of such a search and evaluation of (14) has been compared to our simulations, and is shown as dotted lines in the figures of the next section. The approximate bound is reasonably accurate for higher SNR where neglecting higher weight (and hence higher distance) events is less detrimental to the approximate bound.

We now present a method of interleaver construction that works as follows:

1. Begin with any randomly chosen interleaver.
2. Look for the pattern in the information bit sequence resulting in the smallest distance from the all zero codeword.
3. In this pattern we locate the first position in the information sequence $\vec{d}$ that is involved, in the example of Fig. 3 the position i. We then randomly select another position $\in \{1,...N\}$ but not equal to i, in our example k.
4. Then we simply swap I_i and I_k.
5. Continue with step 2.

Note that the swapping can result in an even worse pattern being created, but after many such operations the interleaver will become better. To reduce the computational burden of step 2, we can initially limit the search to code words of information bit weight 1 (edge effects) or 2, then when we have exceeded a certain minimum distance, continue to include higher weights in the search. After each swapping operation, we need to start our search again at the smallest weight.

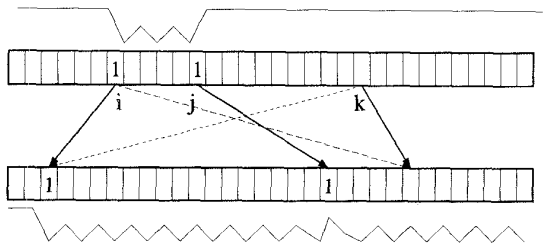

Figure 3: Resolving the event of Fig. 2. I_k and I_i are swapped, (k chosen at random.) Dashed: before swapping.

Recent investigations have shown that the component code with the generators 23 and 35 lead to better distance properties of the turbo-code. The above described weight two and weight four events have a much higher distance. Through our interleaver search, we have found a code that has 7 weight four code words at Hamming distance 21 from the zero code word. However, information bit weight five and higher code words still need to be analyzed.

6 Conclusions

We have shown how the iterative 'turbo' decoder can be formulated without needing an estimate of the variance of any of the decoder outputs, furthermore, we have not found any heuristically determined correction factors that help stability to be necessary as suggested in [3]. The results claimed in [3] (for $N = 65536$) can be reproduced except for a flattening of the BER curves -especially for smaller block sizes.

It is possible to analyze the performance of the turbo-codes at high SNR by giving an approximate union bound for the bit error rate. This involves partial analysis of the interleaver, component codes and puncturing. Extensive search for higher information weight code words remains a challenging task. An interleaver modification procedure was presented that minimizes this approximate bound and has resulted in an improvement where the BER curve flattens out. Finally, we have given a simple suboptimal solution to the trellis-termination problem that remains unmentioned in [3] and show that the degradation due to not terminating the second trellis is only significant for smaller block sizes.

7 Acknowledgements

The author would especially like to thank Dr. J. Lodge, Dr. J. Hagenauer and M. Moher for many invaluable discussions and comments.

References

[1] G. Battail, "Building long codes by combination of simple ones, thanks to weighted- output decoding," in *Proc. URSI*

'89, pp. 634–637, 1989.

[2] J. Lodge, R. Young, P. Hoeher, and J. Hagenauer, "Separable MAP 'filters' for the decoding of product and concatenated codes," in *Proc. ICC'93*, pp. 1740–1745, May 1993.

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes (1)," in *Proc. ICC '93*, May 1993.

[4] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. IT.*, vol. 20, pp. 284–287, March 1974.

[5] G. D. Forney, "The Viterbi algorithm," *IEEE Trans. Commun.*, vol. 61, pp. 268–278, March 1973.

[6] J. Lodge. Personal Communication, June 1993.

[7] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding.* New York: McGraw Hill Book Co., 1979.

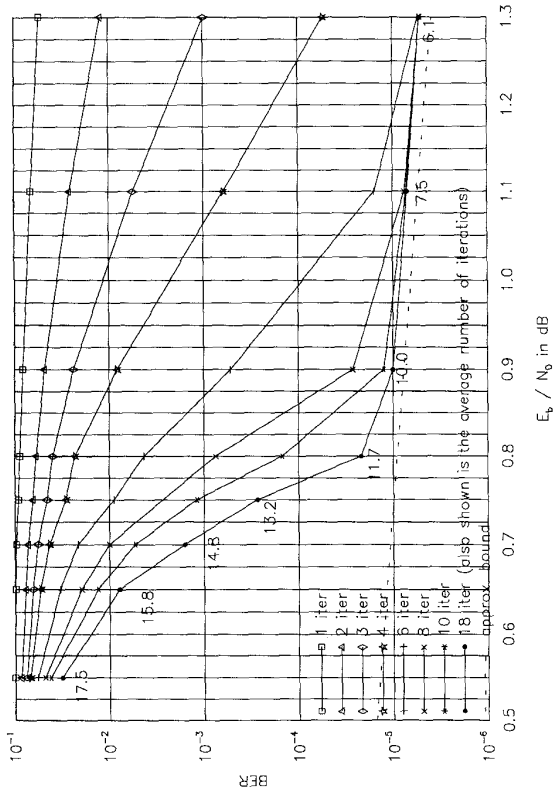

Figure 4: BER of Turbo codes for $N = 10000$, $M1 = M2 = 4$. Generators 37, 21. Dotted line shows approximate bound.

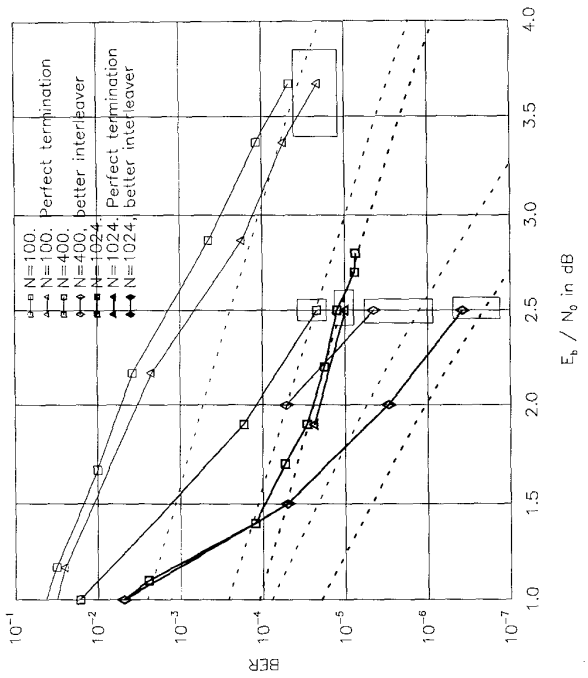

Figure 5: BER of turbo-codes for smaller N and different interleavers. $M1 = M2 = 4$. Generators 37, 21. Dotted lines show approximate bound.

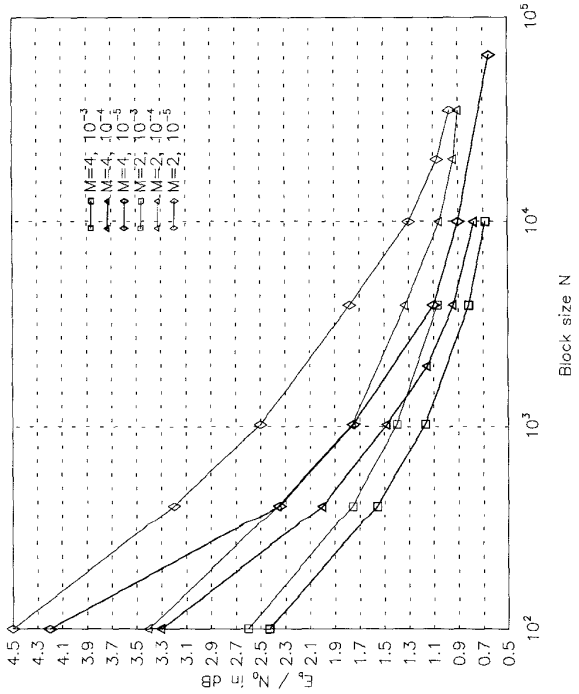

Figure 6: Influence of the block size N on the SNR needed to achieve a BER of 10^{-3}, 10^{-4} and 10^{-5} for component code memory 2 (Generators 5, 7) and memory 4.