REPORTS
IN
INFORMATICS

ISSN 0333-3590

Iterative decoding for the asymmetric channel

Pål Ellingsen
The Selmer Center
University of Bergen
e-mail: paale@ii.uib.no

REPORT NO 295 April 2005

Department of Informatics

UNIVERSITY OF BERGEN

Bergen, Norway

This report has URL `http://www.ii.uib.no/publikasjoner/texrap/ps/2005-295.ps`

Reports in Informatics from Department of Informatics, University of Bergen, Norway, is available at `http://www.ii.uib.no/publikasjoner/texrap/`.

Requests for paper copies of this report can be sent to:

Department of Informatics, University of Bergen, Høyteknologisenteret,
P.O. Box 7800, N-5020 Bergen, Norway

Iterative decoding for the asymmetric channel

Pål Ellingsen*
The Selmer Center
University of Bergen
e-mail: paale@ii.uib.no

21st April 2005

Abstract

We discuss several aspects of coding for the asymmetric channel, with emphasis on iterative coding. We implement turbo- and LDPC decoders for the Z-channel, and a novel coding scheme designed to approach the optimum input distribution for the channel.

Index terms - Z-channel, iterative decoding, channel capacity.

1 Introduction

Often, a symmetric channel model is the appropriate description of an actual physical channel. However there are also situations where an *asymmetric* model is best suited to describe a physical channel. For example, the photon channel could be described by the Z-channel as a photon on the channel might disappear, while no new photons are created by the channel. The same also applies to the effect of gamma-radiation on some types of magnetic memory.
The asymmetric channel is a special case of the general binary channel, in which the transition probability of one of the inputs, say 1, is negligible, i.e. $p(0|1) \neq 0$ Classical codes for the asymmetric channel has been developed before (Refer to [1] for a summary of some important codes). Many codes have been designed for this channel, but so far we have not seen any attempts to use the strengths of iterative decoding techniques to invent good codes for the Z-channel, so that is our aim.

2 The Z-channel

2.1 Channel model

The asymmetric channel, also called the Z-channel, is a discrete, memoryless channel, in which the transition probability of one of the inputs, say 0, is negligible, i.e. $p(1|0) = 0$. Classical algebraic codes for the asymmetric channel have been developed (See [1] for a summary of some important code constructions), and there has also been recent attempts to find good codes for this channel using a feedback channel [2]. We will use the generic model below to model any encoder-decoder system, and also use the notation that X is the input to the channel and Y is the output from the channel. For a code with blocks of length n we will write $X = \{x_1, x_2, \ldots, x_n\}$ and similarly $Y = \{y_1, y_2, \ldots, y_n\}$.

*This research was supported by the RCN (Research Council of Norway) under grant 157668/432

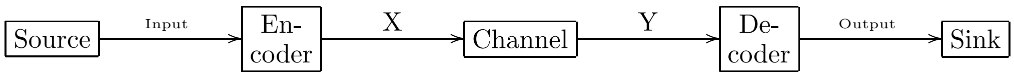
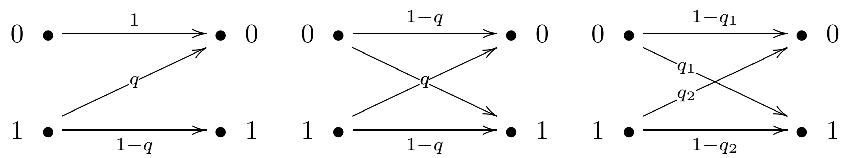


Figure 1: A generic encoder-decoder system.

In the following we will study the Z-channel, which can be seen as a special case of the generalized asymmetric channel when the $0 \to 1$ transition probability q_1 goes to 0.



(a) The Z-channel

(b) The binary symmetric channel

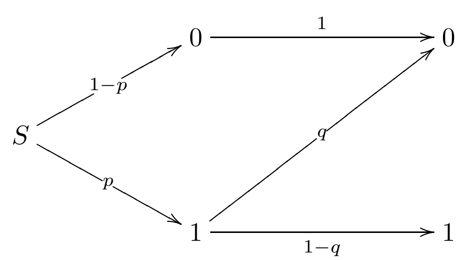
(c) The generalized asymmetric channel

Figure 2: Channel models

2.2 Capacity and input distribution

Since the transition probabilities of the asymmetric channel are non-equal, the capacity not only depends on the probability of bit errors on the channel, but also on the input bit probability to the channel.

We denote the input probability to the channel by $P(x_i = 0) = 1-p$ and $P(x_i = 1) = p$, and let the crossover probability $P(y_i = 0|x_i = 1)$ be q. Thus, the probabilities in the channel model are:



Let the *entropy* of a binary variable X be

$$h(X) = -\sum_{x=0}^{1} p(X = x) \log_2 p(X = x).$$

If $p(X = 1) = p$, then

$$h(X) = -p \log_2 p - (1 - p) \log_2(1 - p).$$

If the distribution of the variable is given as above, we will also write

$$h(X) = h(p) = h(1 - p)$$

The *mutual information* $I(X; Y)$ between two variables X and Y is defined as the difference between the entropy of the first variable and the entropy of the variable conditioned on the second variable, i.e. $I(X; Y) = h(X) - h(X|Y) = h(Y) - h(Y|X)$.

By Shannon [3], the capacity of the channel is given by choosing the input distribution that maximizes the mutual information between the input and output of the channel:

$$C = \max_p I(X;Y)$$

Then, the capacity of the channel is given by

$$\begin{aligned} C = \max_p I(X;Y) &= \max_p (h(Y) - h(Y \mid X) \\ &= \max_p (h(Y) - \sum_{x=0}^{1} h(Y \mid X = x) P(X = x) \\ &= \max_p (h(Y) - h(Y \mid X = 1) P(X = 1)) \end{aligned}$$

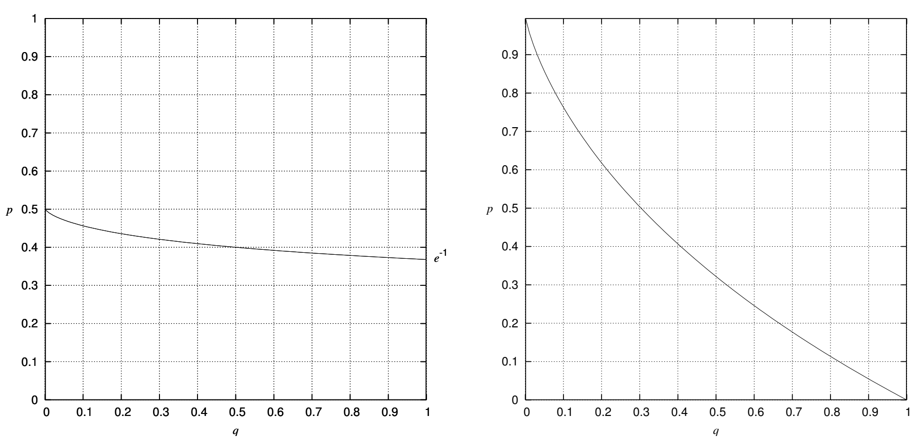
Since $P(Y = 1) = p(1 - q)$ and $P(Y = 1 \mid X = 1) = 1 - q$

$$\begin{aligned} C = \max_p I(X;Y) &= \max_p (-\sum_{Y=0}^{1} P(Y) \log_2 P(Y) && (1) \\ &\quad - \sum_{Y=0}^{1} P(Y \mid X = 1) \log_2 P(Y \mid X = 1) P(X = 1) && (2) \\ &= \max_p (h(p(1 - q)) - h(q)p) && (3) \end{aligned}$$

Given this expression for the capacity, we can take the derivative of (3) with respect to p and set to zero to find the capacity-achieving value of p as a function of q. See Appendix A for details. This yields

$$p = \frac{1}{(1 - q)(2^{\frac{h(q)}{1-q}} + 1)} \qquad (4)$$

We can then find the limits $\lim_{q \to 0} p$ and $\lim_{q \to 1} p$. It is easy to see that $\lim_{q \to 0} p = 1/2$, and by Appendix A, $\lim_{q \to 1} p = 1/e$.



(a) Optimum p, from (4)

(b) Capacity, (4) substituted into (3)

Figure 3: Capacity and optimum p

Plotting (4) in Figure 3, we see that the capacity achieving input distribution for the error free channel tends to 1/2 when the error probability goes to 0, and decreases to e^{-1} as the error probability goes to 1. The capacity of the channel can only be achieved when the rate of 0s from the encoder is slightly larger than the rate of 1s, i. e. using a non-even input distribution to the channel.

3 Iterative decoding for the Z-channel

Prior to the discovery of iterative decoding, the best known codes could not approach the channel capacity as stated in Shannon's Channel Coding theorem. The introduction of turbo codes, LDPC codes, Repeat-Accumulate codes and other iterative message-passing decoding methods made it possible to achieve decoding performance that was very close to the Shannon limit. One of the best known results is an LDPC code-construction by Chung *et al.* [4] that gets within 0.0045 dB of the Shannon limit on an AWGN channel. As far as non-symmetric channels go, McEliece conjectured [5] that turbo-like codes should be effective also on this type of channels. This conjecture is supported by an old theorem of Silverman and Rumsey [6] saying that the ratio of the even weight input capacity to the true capacity is at least $(e/2)\ln 2 = 0.9421$. We will first look at the performance of some linear code-constructions and a bound on their performance on the Z-channel, and we will then investigate a non-linear code and compare its performance to the linear codes.

3.1 Performance bounds of iterative decoding

The bit and frame error rate of a code over the asymmetric channel can be accurately determined by computer simulation as long as the error probability is above the waterfall region for the code in question. It is however, more difficult to determine the performance of the code as the error probability drops beyond this region. The system's behavior cannot easily be simulated due to the extremely low number of decoder errors over this kind of channel. It is, however, possible to give an upper bound of the bit and frame error rates by using the Union Bound.

Theorem 3.1 (Union bound). Let $\{E_1, E_2, E_3, \ldots, E_n\}$ be events in a probability space S with probability measure $\mu(\cdot)$. Then

$$\mu\left(\bigcup_{i=1}^{n} E_i\right) \leq \sum_{i=1}^{n} \mu(E_i)$$

The probability of the union of the events is less than or equal to the sum of the probabilities of the individual events.

We want to find a method for calculation of an upper bound to decoder error probability in the case of a linear code, with maximum likelihood decoding. There exists extensive material on such calculations in the case where the actual channel is supposed to be an additive white Gaussian noise channel (see [7]) or other symmetric channels, but little work has been done on non-symmetric channels. We will first look at the general principle, but with the special case of the asymmetric channel (Z-channel) in mind, as we want to apply the principles to this channel.

Definition 3.1 (Pairwise error probability). The probability that some codeword $\mathbf{z}$ is incorrectly decoded into a codeword $\mathbf{x}$ with Hamming distance $d_H(\mathbf{z}, \mathbf{x}) = d$ from $\mathbf{z}$, is denoted $P_d^{(\mathbf{z})}$. For a symmetric channel, this probability is the same for all pairs of codeword with Hamming distance d, and is called the *pairwise error probability* P_d.

We can then apply the union bound principle by computing P_d for all pairs of codewords and sum over them. According to Theorem 3.1, we will then get an upper bound on error probability under maximum likelihood decoding.
Further, we express the weight distribution of a code of length n by a *weight enumerating function* $A(X) = \sum_{d=0}^{n} A_d X^d$ where the coordinates A_d are the number of codewords of weight d in the code.
Then, the problem reduces to summing over the product of the code's weight enumerator coordinates and the corresponding pairwise error probabilities, so that the word error probability P_w is given by

$$P_w \leq \sum_{d=1}^{n} A_d P_d \tag{5}$$

Not considering the weight enumerator of a code, the problem is how to calculate the pairwise error probability in a specific case.
In the AWGN case, a pairwise error probability can be calculated by means of the $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-y^2/2} dy$ error function so that $P_d = Q(\sqrt{2dRE_b/N_o})$ (Ref. [8]). This approach is not feasible in the case of other channels, but for a DMC, we can approach the problem in a similar way by using the Bhattacharya noise parameter γ for the calculation of P_d.

Definition 3.2 (Bhattacharyya noise parameter). Suppose we have a discrete, memoryless channel with input alphabet $\{0,1\}$ and output alphabet Y, then the Bhattacharya parameter γ is defined as

$$\gamma = \sum_{y \in Y} \sqrt{p(y|0)p(y|1)}$$

Example 3.1. *In the case of an asymmetric channel with crossover probability q we have $Y = \{0,1\}$ and we can find the γ parameter as $\gamma = \sqrt{p(0|0)p(0|1)} + \sqrt{p(1|0)p(1|1)} = \sqrt{q}$.*

This parameter can be used to calculate an upper bound on P_w for the asymmetric channel by using an analogy of a theorem due to McEliece [7].

Theorem 3.2. Let C be a binary linear code used on an asymmetric channel with crossover probability q with input alphabet $\{0,1\}$ and output alphabet Y. Let $A(x) = \sum_{i=0}^{n} A_i x^i$ be the weight enumerator of the code. Then if a maximum likelihood decoding rule is used, an upper bound on word error probability is given by

$$P_w \leq A(\gamma) - 1 \tag{6}$$

Proof. Let P_d be the pairwise error probability, given that $\mathbf{z}$ is transmitted, and let $Y_j = \{\mathbf{y} : p(\mathbf{y}|\mathbf{x}_j) \geq p(\mathbf{y}|\mathbf{z})\}$ i.e. Y_j is the collection of received words that decode into $\mathbf{x}_j$ rather than $\mathbf{z}$.
Then

$$P_d = \sum_{\mathbf{y} \in Y_j} p(\mathbf{y}|\mathbf{z}). \tag{7}$$

Since $\sqrt{p(\mathbf{y}|\mathbf{x}_j)/p(\mathbf{y}|\mathbf{z})} \geq 1$ for all $\mathbf{y} \in Y_j$, we can multiply each term in (7) by this factor and get

$$P_d \leq \sum_{\mathbf{y} \in Y_j} \sqrt{p(\mathbf{y}|\mathbf{z})p(\mathbf{y}|\mathbf{x}_j)}. \tag{8}$$

By extending this summation to all $\mathbf{y} \in Y^n$ we get a somewhat weaker upper bound on P_d.

$$P_d \leq \sum_{\mathbf{y} \in Y^n} \sqrt{p(\mathbf{y}|\mathbf{z})p(\mathbf{y}|\mathbf{x}_j)}. \tag{9}$$

The expansion $p(\mathbf{y}|\mathbf{x}) = p(y_1|x_1)p(y_2|x_2)\ldots p(y_n|x_n)$ yields

$$\sum_{\mathbf{y} \in Y^n} \sqrt{p(\mathbf{y}|\mathbf{z})p(\mathbf{y}|\mathbf{x}_j)} = \sum_{\mathbf{y} \in Y^n} \sqrt{\prod_{i=1}^{n} p(y_i|z_i)p(y_i|x_{ji})}$$

and further

$$\sum_{\mathbf{y} \in Y^n} \sqrt{\prod_{i=1}^{n} p(y_i|z_i)p(y_i|x_{ji})} \leq \prod_{i=1}^{n} \sum_{\mathbf{y} \in Y} \sqrt{p(y_i|z_i)p(y_i|x_{ji})}.$$

The inner sum is clearly 1 if $z_i = x_{ji}$ and γ if $z_i \neq x_{ji}$. Thus, (8) reduces to

$$P_d \leq \gamma^{d_H(\mathbf{z}, \mathbf{x}_j)}. \tag{10}$$

Substituting for P_d in (5) we get

$$P_w \leq \sum_{d=1}^{n} A_d^{(z)} \gamma^d \tag{11}$$

where $A_d^{(z)}$ is the number of codewords at distance d from $\mathbf{z}$. Because of the linearity of the code we get $A_d^{(z)} = A_d$, so finally we get

$$P_w \leq A(\gamma) - 1 \tag{12}$$

□

Since $0 \leq \gamma \leq 1$, γ^d decreases exponentially in d, and therefore the sum in (12) will be dominated by the lower weight terms. Thus, for smaller values of q, it is possible to compute an approximate upper bound without knowing the full weight distribution of the code.

We can extend this further to calculate a similar bound on bit error probability. First, for a code with information length k and total length n, we define an *input-output weight enumerating function* to be a function of two variables $A(W, X) = \sum_{w=0}^{k} \sum_{d=w}^{n} A_{w,d} W^w X^d$ where $A_{w,d}$ is the number of words with information weight w and total weight d.

Theorem 3.3. Let $A_{w,d}$ be the coefficients of the input-output weight enumerator of a code. Then an upper bound on the bit error rate of the code under maximum likelihood decoding is given by

$$P_b \leq \sum_{d=1}^{n} \sum_{w=0}^{k} \frac{w}{k} A_{w,d} \gamma^d \tag{13}$$

Proof. If a codeword with information weight w is incorrectly decoded, this can at most result in a total of w bit errors out of the k info bits. Hence the bit error probability for the codeword is bounded above by $\frac{w}{k}$. The probability of such a decoding error is given by $A_{w,d}\gamma^d$, and the bit error probability becomes $\frac{w}{k} A_{w,d}\gamma^d$ so we get the total bit error probability by summing over these terms. □

3.2 Linear codes

As the upper bound on ML-decoding applies to linear codes, we shall study a couple of examples of such codes and see how well they work on the asymmetric channel. A short description of the classical turbo- and LDPC codes is given, the necessary modifications to the decoders for the channel are presented, and finally simulation results of an implementation of an instance of each code are compared to the bound found in section 3.1.

3.2.1 Turbo codes

A standard concatenated encoder for the asymmetric channel consist of two component encoders and a multiplexer, as the standard parallel concatenated convolutional encoder.

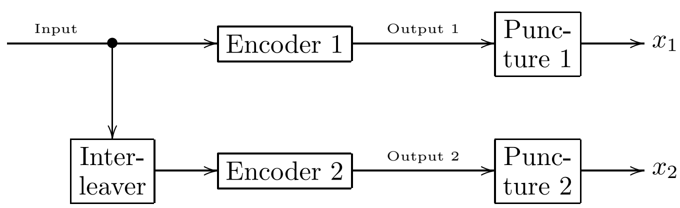
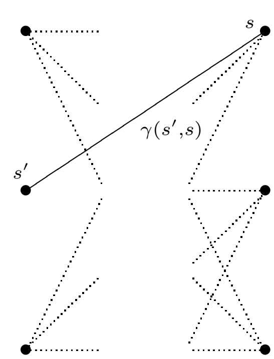


Figure 4: A generic parallel concatenated encoder.

The decoder for the system is a log-MAP turbo decoder, which is adapted to the channel by using branch metrics that reflect the nature of the channel. The channel uses bipolar modulation of the binary input vector so that $1 \to -1.0$ and $0 \to 1.0$. This choice of modulation seems natural as it preserves the input's structure viewed as the group $\mathbb{Z}_2$.
If we assume that the BCJR algorithm is used in the decoders, the generic MAP equation describing the trellis is

$$\alpha_i(s') \cdot \gamma_i(s', s) \cdot \beta_{i+1}(s)$$

where α is the forward metric, *gamma* is the branch metric and β is the backward metric.



The branch metric γ_i can be written

$$\gamma_i(s, s') = P(u_k)P(y_k \mid u_k)$$

For the asymmetric channel, the conditional probabilities are given by

Branch label (u_k)	Received value (y_k)	$P(y_k \mid u_k)$
0	1.0	1
0	-1.0	q
1	1.0	0
1	-1.0	$1-q$

Table 1: Branch metrics

3.2.2 Results

We use a pair of rate 8/9 parallel concatenated convolutional codes punctured to rate 8/10 over an asymmetric channel. Assuming the input to the channel is distributed as $p(1) = p(0) = \frac{1}{2}$, the Shannon bound, which gives the upper bound on the crossover probability q for error free decoding, is $q = 0.077$. The information length is 1280 bits. The simulation results compared to the upper bound on the ML error floor from (12) are shown in Figure 5. For higher values of q, the decoding does not approximate ML-decoding, but as the decoding converges to ML-decoding below the waterfall region, we observe that the predicted upper bound indeed seems to upper limit the performance of the decoder.

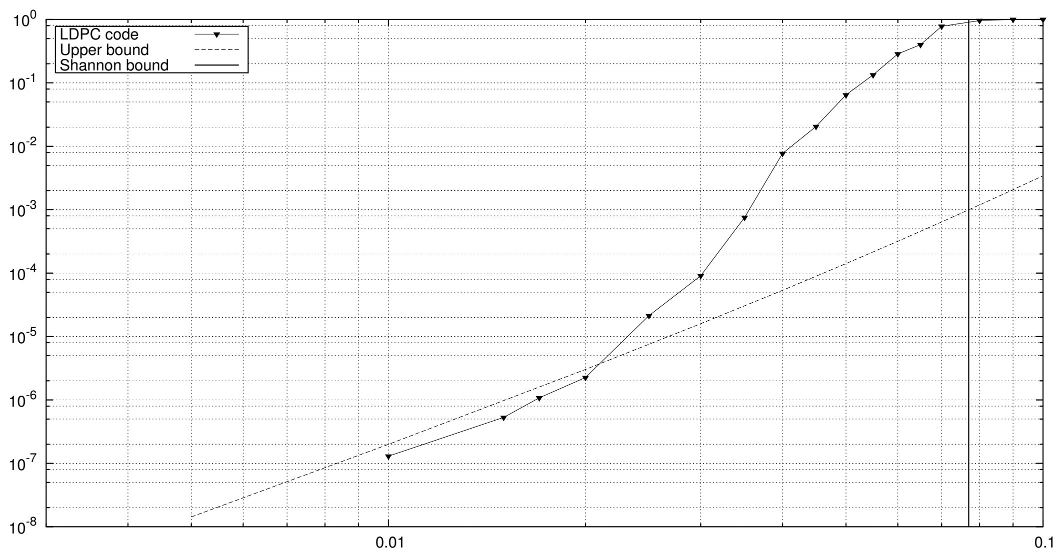


Figure 5: Frame error rate

3.2.3 LDPC-codes

LDPC codes is another class of codes that uses belief propagation to attain near-capacity decoding. The codes are sparse linear block codes that may be pseudorandom or result of an explicit construction. The code may be represented as a Tanner graph (see Fig. 6) where the parity checks of are represented by ⊞ and the variable nodes are represented by ◯. The decoding can be viewed as message passing on the same graph. Initially each variable nodes send messages to its parity check nodes indicating the probability of it being a $+1$ versus a -1. The check nodes returns the probabilities from all its neighbors, except from the node itself. The subsequent iterations proceeds analogously, except for that the information passed form the variable nodes is based on both the channel values and the information received in the previous iteration.

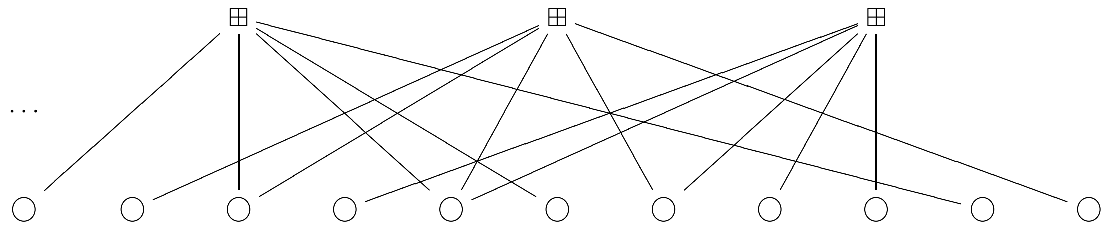


Figure 6: Tanner graph

3.2.4 Results

We use a regular, linear block code with codelength $n = 495$ and information length $k = 433$ due to MacKay [9]. The lower part of the weight spectrum for this code was found by Hu and Fossorier in [10]. For a linear code of rate $\frac{433}{495}$, the Shannon bound is $q = 0.041$. Simulation results, compared to the union bound based upper limit are shown in Figure 7. We observe as in Figure 5 that as the decoding performance approaches ML- decoding, the predicted upper bound again seems to be correct.

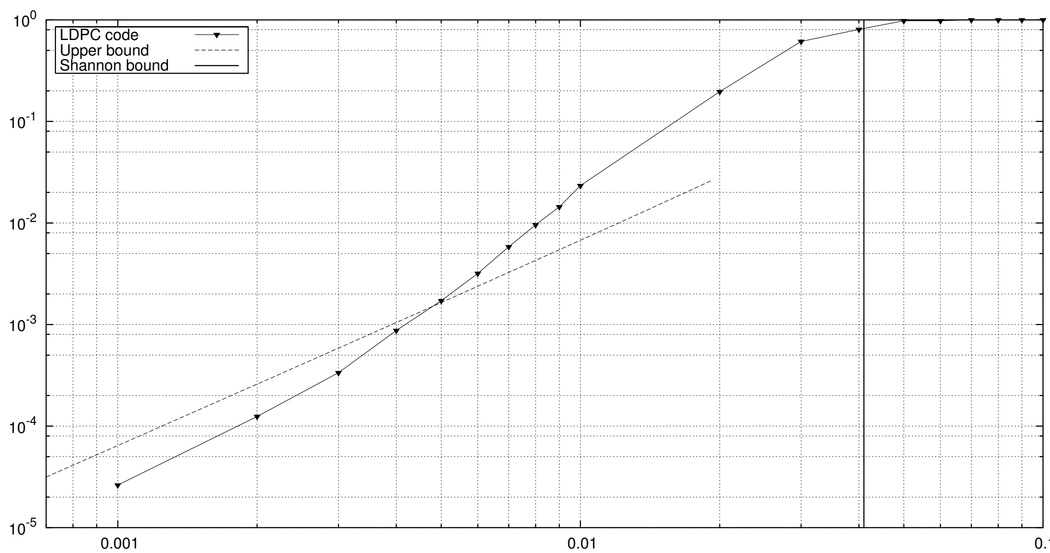
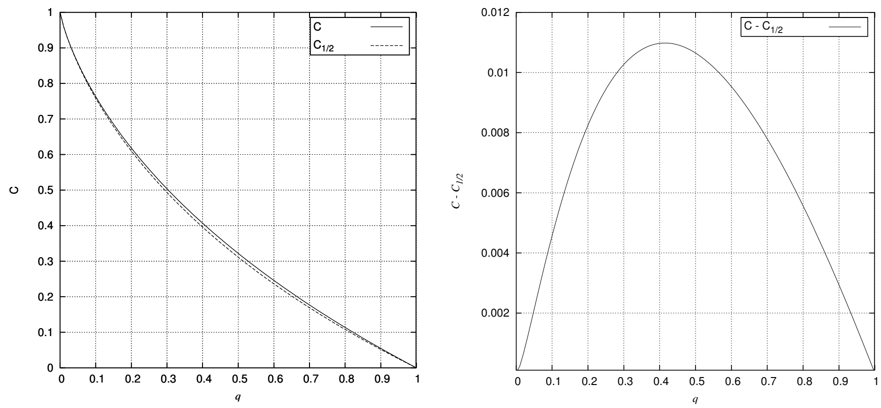


Figure 7: Frame error rate

3.3 Nonlinear codes

As became apparent from the discussion in chapter 2.2, capacity can only be achieved on the Z-channel with a uneven input distribution. We will therefore investigate the use of a code consisting of both linear and nonlinear components, to approach the optimum input distribution and thereby achieve good performance on the Z-channel. As we have seen, codes with an even input distribution can also perform quite well on this channel for certain values of q, and we would like to determine the range where we have the most gain from using an uneven input to the channel. The "capacity" for a linear code on the Z-channel can be determined by taking $C_{\frac{1}{2}} = I(\frac{1}{2}; q) = h(\frac{1}{2}q) - \frac{1}{2}h(q)$, and as we see in Fig. 8(a)

Taking the difference between this and the true capacity, we see which values of q gives the biggest gain from using a code with the optimum input distribution.



(a) C and $C_{1/2}$

(b) Difference between C and $C_{1/2}$

Figure 8: Difference between even and optimum input distribution

We see clearly that the most interesting range of values of q are centered around 0.4, with the difference between the two quantities being very small near the ends.

3.3.1 Choice of component codes

Given the very good performance of LDPC-codes on the standard symmetric channel, and the fact that a linear code would be able achieve at least 94% of capacity, which means error free decoding at $\Delta q = 0.012$ from the capacity of the Z-channel, it seems reasonable to try to combine a regular LDPC-code with a nonlinear code that skew the weight distribution in the desired direction.
Using a combination of a rate 3/4 nonlinear code and a rate 4/9 LDPC-code, we get a non-linear code that are comparable to rate 1/3 LDPC-codes.
For the nonlinear component, if we concentrate on the range of values mentioned at the end of section 2.2, we can examine some values of q to see what the typical optimum input distribution for these values are, and the corresponding capacities.

q	$p(q)$	C
0.1	0.456	0.763
0.2	0.436	0.618
0.3	0.421	0.504
0.4	0.409	0.407
0.5	0.4	0.322

(14)

As we want to keep the rate of the distribution shaping code as high as possible in order to let the LDPC-component be as long as possible, we will look at nonlinear code components of rate $\frac{i}{i+1}$.
To pick the actual code from among the many possible ones, we must consider the property of the component code that are most interesting to us, namely its weight distribution. In general, the codes of rate $\frac{i}{i+1}$ where the check bit is 0 except for one case, for instance

$$\begin{array}{ccc} 00 & \cdots & 00 \\ 00 & \cdots & 10 \\ & \vdots & \\ 11 & \cdots & 11 \end{array}$$

will have the weight distribution

$$w(i) = \frac{i2^{i-1}+1}{(i+1)2^i}.$$

We see that $\lim_{i\to\infty} w(i) = \frac{1}{2}$ so we obviously do not want large values of i.
For some small values of i we get

i	$w(i)$
2	0.417
3	0.406
4	0.413
5	0.422

(15)

We observe that the $\frac{i}{i+1}$ codes with $2 \leq i \leq 5$ has has the most desirable input distribution for use with an asymmetric channel with error probability in the range from 0.3 to 0.5.
For the simulations, we choose to study a code with overall rate 1/3 since this rate corresponds to the capacity of a channel with q between 0.4 and 0.5 as seen in table 14. We obtain this rate by combining a rate 3/4 nonlinear code of the kind described above, with a rate 4/9 LDPC code.

3.3.2 Encoder / decoder setup

The idea is to combine the good error correcting capabilities of the LDPC code with a nonlinear code to achieve a better input distribution for the channel. The ordering of the components is crucial to the code's performance. If we first encode the information bits with the LDPC code, and then with the nonlinear code, the total distribution of the code will be that of the outer, nonlinear code, and hence close to the optimum value. However, then we would not get any gain from decoding the nonlinear code after the first iteration, since none of its check nodes would be involved in other checks that could produce a coding gain by iterating on the received information.
If, on the other hand, we use the LDPC code as the outer code, the code will be better suited for iterative decoding as all bits now can get an information gain as the decoder iterates on the received information, but the resulting code will have a distribution that is closer to 0.5.
Thus, we will use a code setup shown in Figure 9. The information block is first encoded using the nonlinear code component, then the resulting block is interleaved and encoded by the LDPC encoder. The decoding process starts by decoding the part of the code that was encoded by the NL-encoder; the output of this decoding is then deinterleaved and merged with the channel values, and sent to the LDPC-decoder. The decoder will iterate until a codeword is found or the maximum number of iterations are reached. We must consider the rate of the outer encoder to find the real input distribution. If the rate of the nonlinear encoder component is k_1/k_2 and the rate of the LDPC encoder is n_1/n_2, the total rate is k_1n_1/k_2n_2. If we

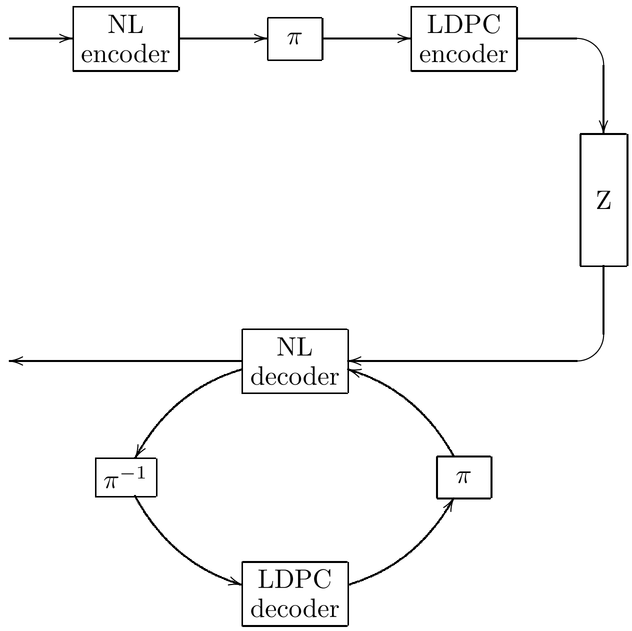


Figure 9: Code setup

let the weight distributions of the two component codes be p_1 and p_2, the input distribution is then given by

$$\frac{n_1 p_1}{n_2} + \frac{(n_2 - n_1) p_2}{n_2}.$$

Assuming we use a code with overall rate 1/3, the code obtained when using the LDPC-code as outer code will have distribution $0.4 \cdot \frac{4}{9} + 0.5 \cdot \frac{5}{9} = \frac{41}{90} \approx 0.456$. The decoder will then be able to do error free decoding up to $q = 0.482$, while a code of rate 1/3 could do error free decoding up to $q = 0.486$ if the input had the optimum distribution of 0.401.
Consequently, we will get a code that can not achieve capacity, based on the requirements presented in the previous chapter. However, since the code is capable of error free decoding for a q up to 0.482, at least in theory it can do better than a linear code, for which the limit is 0.473.

3.3.3 Trellis description of the 3/4 majority decision code

In order to do effective iterative decoding of the code described above, we need to find a SISO decoder for the nonlinear rate 3/4 code. We will implement this decoder as a trellis decoder

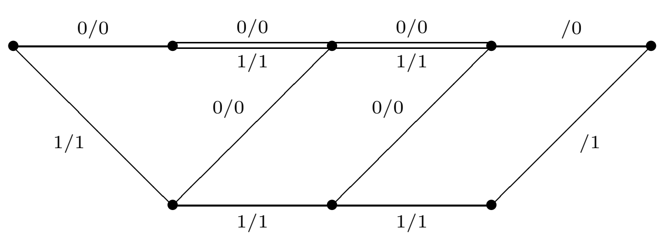


Figure 10: Trellis for nonlinear code

This leads to the trellis in Figure 10, where the states indicate the concurrent check bit state, and the labels on the edges indicate the corresponding input and output values to the trellis.

3.3.4 Factor graph for the nonlinear code

The information bits are encoded with the rate 3/4 nonlinear code. These checks are denoted $\boxdot$. The LDPC component with rate 4/9 then encodes the resulting word; these checks are denoted $\boxplus$, also adding its own check nodes ⓒ to the word. The factor graph of the resulting code then becomes an extension of the classical Tanner graph for the LDPC code component.

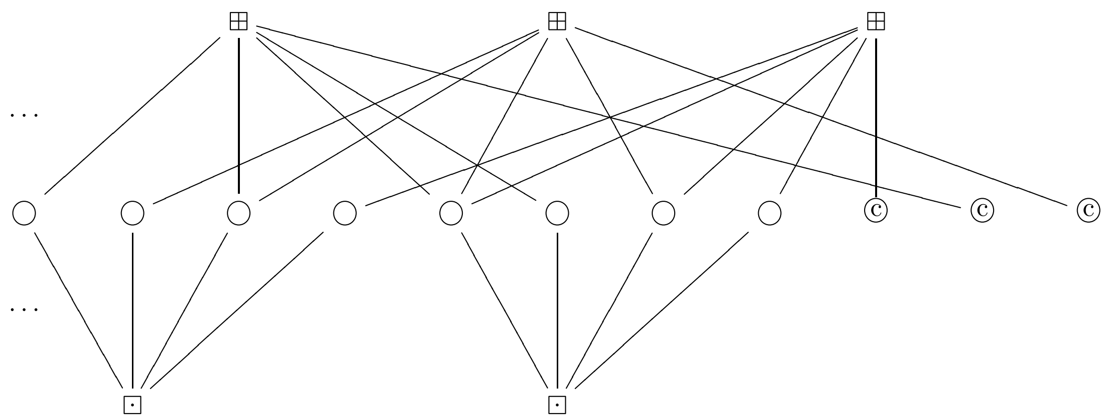


Figure 11: Factor graph for nonlinear code

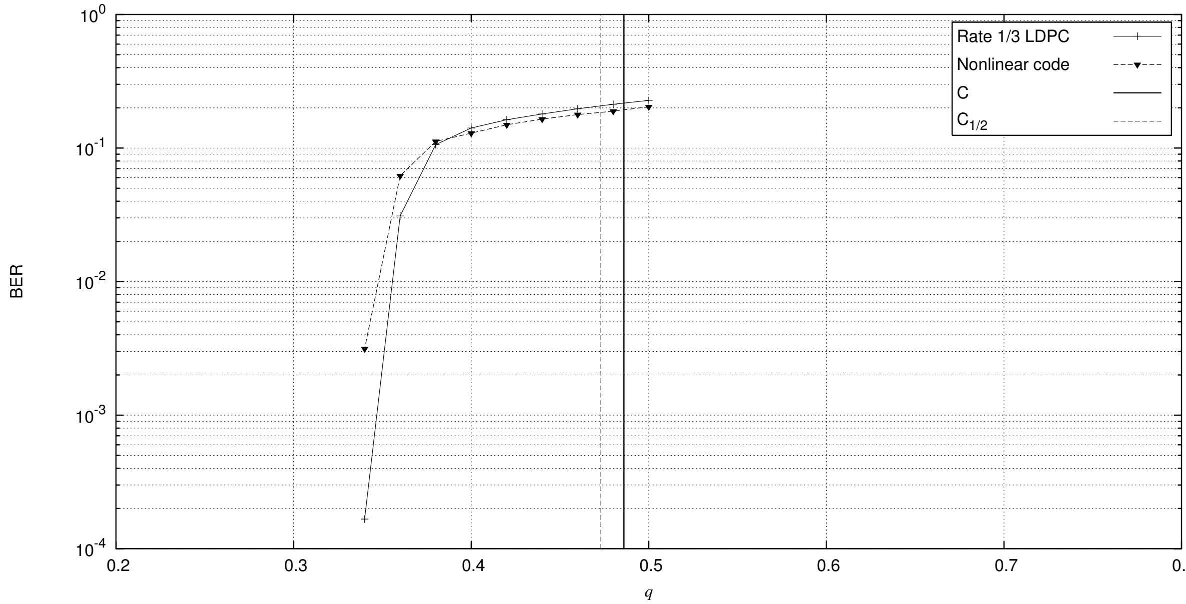
In the graph sketched above information is sent alternatingly from the nonlinear and the LDPC-check nodes to the variable nodes in the middle.

3.3.5 Results

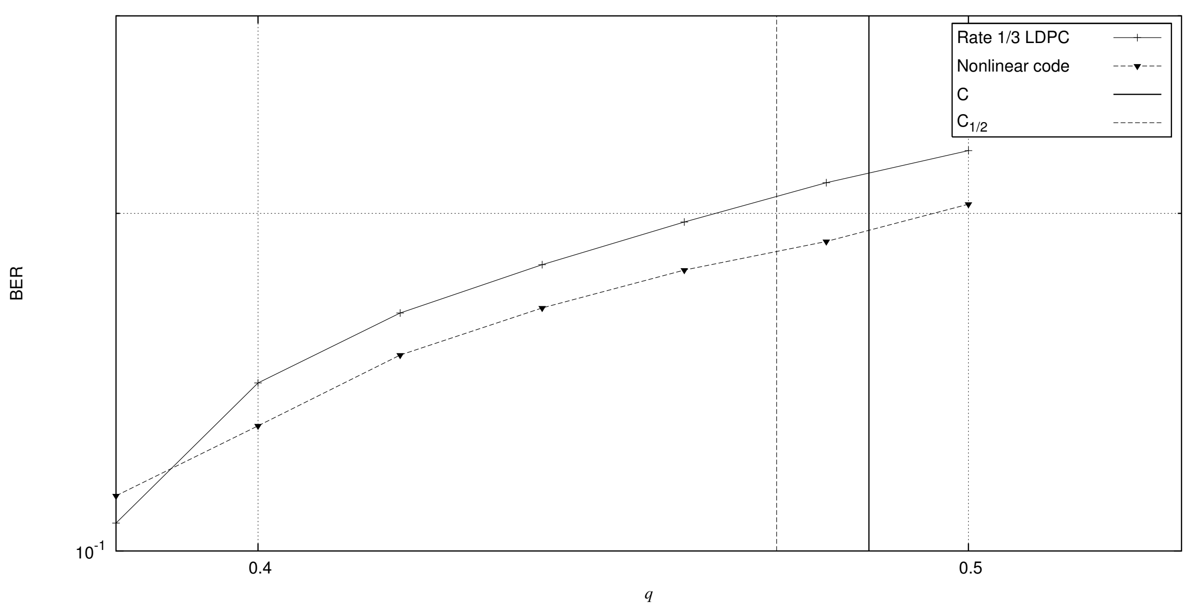
We can see the results of this decoding compared to other codes of different length. Most interesting is the comparison with a full rate 1/3 LDPC code. In the plots, the notation $C_{1/2}$ and C is used for the Shannon limit of a linear and an optimal code, respectively. We observe in Figure 12(a) that the LDPC code has somewhat better performance in the waterfall region while the nonlinear code has better performance in the region above. Detail of the plot in the region above the waterfall is shown in Figure 12(b), and while the differences may seem small one should keep in mind that the Shannon limit of the two codes only differ by $\Delta q = 0.009$.

4 Conclusions

We have seen that the Z-channel requires an uneven input distribution for any code that can achieve capacity. We have also shown that it is possible to compute the exact value for the optimum input distribution for a given q. On the other hand, we have noted that a linear code can achieve at least 94% of the capacity of the channel for any value of q. We have constructed a nonlinear code that is closer to the optimum input distribution for $q = 0.486$. Compared to a linear code, our construction does somewhat better initially but is eventually outperformed by the linear code. The bottom line seems to be that it is difficult to gain a distinct



(a) Code performance nonlinear vs. linear codes, bit error rates



(b) Magnified area from above figure

Figure 12: Nonlinear code performance

advantage by using nonlinear codes as one would have to squeeze the waterfall region of the code between the limits of C and $C_{1/2}$ in order to demonstrate definite superiority of the nonlinear code. In comparison, constructions by Chung in [4] approaches C on the AWGN channel just within the 94% limit at a BER of 10^{-6}.

Appendix A

The calculation of the capacity for the Z-channel and an expression for the capacity achieving input distribution p.
The channel capacity is given by

$$\begin{aligned} C = \max_p I(X;Y) &= \max_p (h(p(1-q)) - h(q)p) \\ &= \max_p (-p(1-q)\log_2(p(1-q)) \\ &\quad -(1-p(1-q))\log_2(1-p(1-q)) - h(q)p) \end{aligned}$$

To find the capacity achieving input distribution, we take the derivative of the expression for the capacity with respect to p, set it to zero, and solve for p.

$$\frac{\mathrm{d}}{\mathrm{d}p}[-p(1-q)\log_2(p(1-q)) - (1-p(1-q))\log_2(1-p(1-q)) - h(q)p] =$$
$$(1-q)(\log_2 \frac{1-p(1-q)}{p(1-q)}) - h(q)$$

$$\begin{aligned} (1-q)(\log_2 \frac{1-p(1-q)}{p(1-q)}) - h(q) &= 0 \\ (1-q)(\log_2 \frac{1-p(1-q)}{p(1-q)}) &= h(q) \\ \log_2 \frac{1-p(1-q)}{p(1-q)} &= \frac{h(q)}{1-q} \\ 2^{\log_2 \frac{1-p(1-q)}{p(1-q)}} &= 2^{\frac{h(q)}{1-q}} \\ \frac{1-p(1-q)}{p(1-q)} &= 2^{\frac{h(q)}{1-q}} \\ p &= \frac{1}{(1-q)(2^{\frac{h(q)}{1-q}}+1)} \end{aligned}$$

Now we can find the capacity achieving p for all values of $q \in \langle 0, 1 \rangle$. Further, we can find the limits of p when q approaches 0 and 1.

$$\begin{aligned} p_0 &= \lim_{q \to 0} \frac{1}{(1-q)(2^{\frac{h(q)}{1-q}}+1)} \\ &= \lim_{q \to 0} \frac{1}{(2^0+1)} \\ &= \frac{1}{2} \end{aligned}$$

$$
\begin{aligned}
p_1 &= \lim_{q\to 1} \frac{1}{(1-q)(2^{\frac{h(q)}{1-q}}+1)} \\
&= \lim_{q\to 1} \frac{1}{(1-q)(2^{\frac{-q\log_2 q-(1-q)\log_2(1-q)}{1-q}}+1)} \\
&= \lim_{q\to 1} \frac{1}{(1-q)(2^{\frac{-q}{1-q}\log_2 q-\log_2(1-q)}+1)} \\
&= \lim_{q\to 1} \frac{1}{(1-q)(\frac{q^{\frac{-q}{1-q}}}{(1-q)}+1)} \\
&= \lim_{q\to 1} \frac{1}{q^{\frac{q}{q-1}}+1-q} \\
&= \lim_{q\to 1} \frac{1}{q^1 q^{\frac{1}{q-1}}+1-q}
\end{aligned}
$$

Let $t+1=q$

$$
\begin{aligned}
&= \lim_{t\to 0} \frac{1}{(t+1)(t+1)^{\frac{1}{t}}-t} \\
&= \frac{1}{e}
\end{aligned}
$$

References

[1] T. Kløve, *Error Correcting Codes for the Asymmetric Channel.* HiB, N-5020 Bergen, Norway: Department of Informatics, University of Bergen, 1995.

[2] L. G. Tallini, S. Al-Bassam, and B. Bose, "On the capacity and codes for the z-channel," in *Proc. IEEE International Symposium on Information Theory*, 2002, p. 406.

[3] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 397–423 and 623–656, July and October 1948.

[4] S.-Y. Chung, G. D. Forney Jr., T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 db of the shannon limit," *IEEE Communications Letters*, vol. 5, no. 2, pp. 58–60, February 2001.

[5] R. J. McEliece, "Are turbo-like codes effective on nonstandard channels?" *IEEE Information Theory Newsletter*, vol. 51, no. 4, pp. 1–8, 2001.

[6] R. A. Silverman, "On binary channels and their cascades," *IEEE Transactions on Information Theory*, vol. 1, no. 3, pp. 19–27, 1955.

[7] R. J. McEliece, *The Theory of Information and Coding*, M. Kac, Ed. Addison-Wesley, 1977.

[8] S. Lin and D. CostelloJr., *Error Control Coding.* Upper Saddle River, NJ07458: Prentice Hall, 2004.

[9] D. J. Mackay, "Encyclopedia of sparse graph codes." [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html

[10] X.-Y. Hu and M. Fossorier, "On the computation of the minimum distance of low-density parity-check codes," 2004. [Online]. Available: www.inference.phy.cam.ac.uk/mackay/codes/MINDIST/main.pdf