

Low-density parity-check codes on Markov channels

Edward A. Rutzer

Cavendish Laboratory, Madingley Road, Cambridge CB3 0HE

Phone: +44 (0) 1223 337344

Email: ear23@mrao.cam.ac.uk

Abstract

Low-density parity-check codes are studied over Markov channels with time-varying gaussian or bit-flipping noise. The belief propagation decoding algorithm is extended to include channel estimation and the gains this extension produce are experimentally determined.

1 Introduction

Low-density parity check (LDPC) codes have been well studied over memoryless channels and are known to have good performance [8]. However the study of them over channels with memory is a less well developed field. Worthen and Stark have described extending belief propagation to include channel inference [14] and experimental work on channels with block memory [13].

Markov channels are a useful bursty channel model as with a sufficient number of states they can model many noise characteristics [11, 15]. However a two-state system shows enough complexity to test codes [9]. Worthen [12] describes some initial work with Markov channels and Wadayama [10] and Garcia-Frias [3] both have presented some results with bit-flipping channels. In this paper we will extend this by looking at both gaussian noise and bit-flipping noise two-state Markov channels.

2 LDPC Codes

LDPC codes are a class of linear block codes. A binary LDPC code is defined by its sparse parity-check matrix, \mathbf{H} , often constructed to have particular column and row weights. Codes constructed in this manner are known to have intrinsically good performance using a maximum likelihood decoder. See [6] for a comprehensive coverage of LDPC codes.

2.1 Decoding on a graph

A parity-check matrix fragment:

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 0 & \cdots \\ 0 & 1 & 1 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \quad (1)$$

will create the bipartite graph fragment shown in figure 1. The columns and rows correspond to symbol nodes and check nodes respectively; the position of 1s denote connections between them. In the following \mathcal{G}_{ij} is the condition that a link exists between check node i and symbol node j (ie iff $H_{ij}=1$).

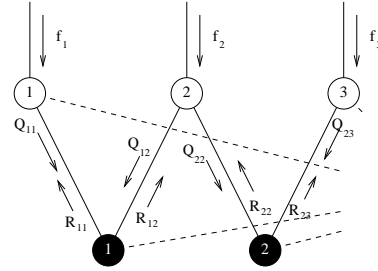


Figure 1: A graph fragment corresponding to Equation 1. Filled and open circles represent the check nodes and symbol nodes respectively. In this example the row and column weights are 3 and 2 respectively

2.2 Check Node Computation

Each check node i sends to symbol node j a message R_{ij}^a that is an approximation to the probability of check i being satisfied if symbol j is a ($a \in \{0, 1\}$):

$$R_{ij}^1 = \Pr(\sum_l H_{il} x_l = 0 | x_j = 1) \quad (2)$$

$$= \sum_{\mathbf{x}: x_j=1} \Pr(\sum_l H_{il} x_l = 0 | \mathbf{x}) \Pr(\mathbf{x} | x_j = 1) \quad (3)$$

$$\approx \sum_{\mathbf{x} \in C_i: x_j=1} \prod_{k: k \neq j, \mathcal{G}_{ik}} Q_{ik}^{x_k} \quad (4)$$

$$R_{ij}^0 = 1 - R_{ij}^1 \quad (5)$$

where \mathbf{x} is the transmitted vector, Q_{ij}^a are the messages received from the symbol nodes (an approximation to the probability that symbol j is a according to the check nodes other than i and the channel probabilities) and C_i is the set of all the valid code words of the check node. Equation 4 can be evaluated by the forward-backward algorithm [1].

2.3 Symbol Node Computation

We need to evaluate both the messages to be passed back to the check nodes Q_{ij}^a and a tentative decoding for that symbol node. Q_{ij}^a is an approximation to the probability that symbol j is a according to the channel probabilities and the check nodes other than i – the symbol S_i is used below to indicate that check node i is satisfied:

$$Q_{ij}^1 = \Pr(x_j = 1 | \{\forall S_k : \mathcal{G}_{kj}, k \neq i\}, \mathbf{f}) \quad (6)$$

$$= Z_{ij} \Pr(x_j = 1 | \mathbf{f}) \Pr(\{\forall S_k : \mathcal{G}_{kj}, k \neq i\} | x_j = 1) \quad (7)$$

$$\approx Z_{ij} f_j(1) \prod_{k: k \neq i, \mathcal{G}_{kj}} R_{kj}^1 \quad (8)$$

$$Q_{ij}^0 = 1 - Q_{ij}^1 \quad (9)$$

where $f_j(a)$ is the prior probability that symbol j is a and Z_{ij} is a normalizing constant.

The tentative decoding for that symbol is:

$$\hat{t}_j = \max_a (\Pr(x_j = a | \{\forall S_k : \mathcal{G}_{k_j}\}, \mathbf{f})) \quad (10)$$

$$\approx \max_a \left(f_j(a) \prod_{k: \mathcal{G}_{k_j}} R_{k_j}^a \right) \quad (11)$$

2.4 Iteration

Q_{ij}^1 is first initialized to $f_i(1)$. Messages are then passed on the bi-partite graph in a chosen manner (typically all the check nodes are updated, followed by all the symbol nodes and this is then repeated) until a successful decoding results or a maximum number of steps has been taken.

3 Markov channels

A Markov channel is defined by a set of states and the transition probabilities between these states. Each state will have a characteristic noise associated with it. In this paper this noise will either be bit-flipping noise with a symmetrical bit-flip probability p_f or gaussian noise with a particular standard deviation. Each bit transmission will have a state associated with it – the channel state for the next transmission is determined by the state transition probabilities. Each state therefore has a typical run length in the state sequence; if a particular state has a probability p_c of changing to any different state then:

$$E(\text{run length}) = \sum_{i=1}^{\infty} i(1-p_c)^{i-1} p_c \quad (12)$$

$$= 1/p_c \quad (13)$$

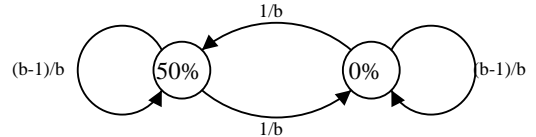
The capacity of the two-state bit-flipping channel (called the Gilbert-Elliott channel) has been calculated in [7]. Using a simple channel model, figure 2(a), the capacity and two related capacities were calculated, figure 2(b):

Gilbert-Elliott capacity This is as calculated by the iterative calculation in [7]. Computationally, 1000 bins were used. It can be seen that for long typical burst lengths the accuracy of the calculation is starting to reduce, causing a deviation from a smooth curve.

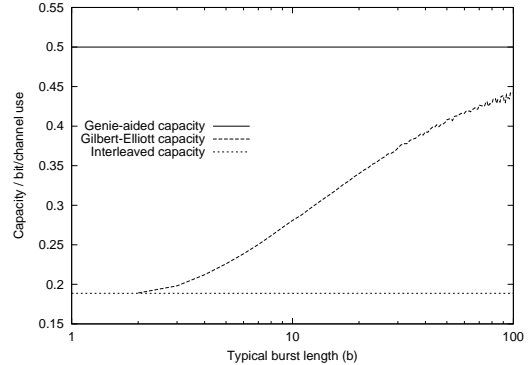
Genie-aided capacity This is the capacity when the state of the channel is available to the decoder. For the example illustrated, the channel then splits into two channels, an erasure channel (when $p_f = 50\%$), and a noiseless channel (when $p_f = 0\%$). The capacity is then 0.5 since each channel is used half the time.

Interleaved capacity If the Markov properties of the channel state are ignored then the channel can be considered as a BSC with noise level p_f equal to the time-average bit flip rate. For the example shown this is $p_f = 25\%$.

When the bursts are very small they are hard to distinguish when decoding and hence as the typical burst length decreases the Gilbert-Elliott capacity tends to the interleaved capacity. Conversely as the burst length increases one can detect more easily where the bursts are and the capacity tends towards the genie-aided capacity.



(a) An example channel model. Each state is represented by a circle indicating p_f and the arrows are labelled with state transition probabilities



(b) Capacity

Figure 2: An example Gilbert-Elliott channel and its capacity

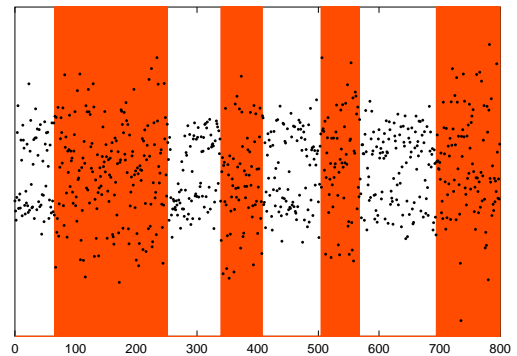


Figure 3: A sample set of received amplitudes from the transmission of i.i.d. bits over a Markov channel with two gaussian noise states separated by 7dB. The transmissions occurring during the noisy state are shaded.

The capacity for gaussian noise channels has not been calculated to our knowledge (more complicated finite alphabet channels are addressed in [4]).

4 Modifications to the decoding algorithm

The most basic decoding algorithm ignores the time-dependent properties of the channel and calculates the input probabilities f_i using the average noise characteristics. We will call this the “No state estimation” algorithm.

An improvement would be to estimate the channel state based on the amplitudes received (an example set of received amplitudes is shown in figure 3). We can use the different characteristic forms of the received signal in each of the states to estimate the state sequence using a standard HMM algorithm, the forward-backward algorithm [2].

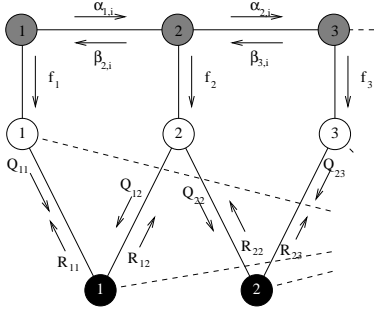


Figure 4: The one-off state estimation algorithm with channel nodes shown in grey

We want to calculate:

$$\Pr(x_i=1|\mathbf{r}) = \sum_{j \in \mathcal{S}} \Pr(x_i=1|\mathbf{r}, s_{ij}) \Pr(s_{ij}|\mathbf{r}) \quad (14)$$

$$= \sum_{j \in \mathcal{S}} \Pr(x_i=1|r_i, s_{ij}) \Pr(s_{ij}|\mathbf{r}) \quad (15)$$

where \mathbf{r} is the received signal, \mathcal{S} is the set of states and s_{ij} represents being in state j at time i . The first term can be calculated using Bayes's theorem and the noise model. The second term can be calculated as follows:

$$\Pr(s_{ij}|\mathbf{r}) \propto \Pr(s_{ij}, \mathbf{r}) \quad (16)$$

$$\propto \Pr(s_{ij}, r_{1,\dots,i}) \Pr(r_{(i+1),\dots,N}|r_{1,\dots,i}, s_{ij}) \quad (17)$$

$$\propto \Pr(s_{ij}, r_{1,\dots,i}) \Pr(r_{(i+1),\dots,N}|s_{ij}) \quad (18)$$

$$\propto \alpha_{ij} \beta_{ij} \text{ by definition} \quad (19)$$

α and β are called the forward and backward probabilities and can be evaluated recursively:

$$\alpha_{ij} = \sum_{k \in \mathcal{S}} \Pr(s_{ij}|s_{i-1,k}) \Pr(r_i|s_{ij}) \alpha_{i-1,k} \quad (20)$$

$$\beta_{ij} = \sum_{k \in \mathcal{S}} \Pr(s_{i+1,k}|s_{ij}) \Pr(r_{i+1}|s_{i+1,k}) \beta_{i+1,k} \quad (21)$$

using the following boundary conditions:

$$\alpha_{0,i} = \Pr(s_{0,i}) \quad (22)$$

$$\beta_{N,i} = 1 \quad (23)$$

This algorithm can be seen as adding extra nodes to the decoding graph which we will call channel nodes, as illustrated in figure 4. We will call the whole decoding algorithm the “one-off state estimation” algorithm. This algorithm will not work for bit-flipping channels as $\Pr(r_i|s_{ij})$ is independent of s_{ij} for i.i.d. transmitted bits.

A further improvement would be to expand belief propagation over these new nodes [14]. In the traditional belief propagation style we will have messages g_i coming up from the symbol nodes:

$$g_i^a = \Pr(x_i = a | \{\forall S_k : \mathcal{G}_{ki}\}) \quad (24)$$

$$= Z_i \Pr(\{\forall S_k : \mathcal{G}_{ki}\} | x_j = a) \quad (25)$$

$$\approx Z_i \prod_k R_{ki}^a \quad (26)$$

and the computation at each node will have to be changed:

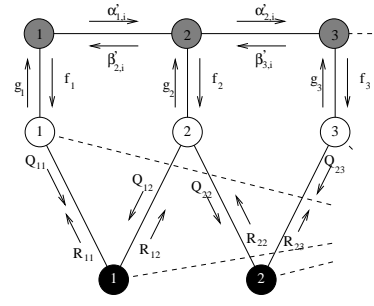


Figure 5: The iterative state estimation decoding algorithm

$$\Pr(x_i=1|\mathbf{r}, \{\forall g_k : k \neq i\})$$

$$= \sum_{j \in \mathcal{S}} \Pr(x_i=1|r_i, s_{ij}) \Pr(s_{ij}|\mathbf{r}, \{\forall g_k : k \neq i\}) \quad (27)$$

$$\propto \sum_{j \in \mathcal{S}} \Pr(x_i=1|r_i, s_{ij}) \alpha'_{ij} \beta'_{ij} \quad (28)$$

where α' and β' are a modified version of α and β to include g :

$$\alpha'_{ij} = \sum_{k \in \mathcal{S}} \Pr(s_{ij}|s_{i-1,k}) \Pr(r_i|s_{ij}, g_i) \alpha_{i-1,k} \quad (29)$$

$$\beta'_{ij} = \sum_{k \in \mathcal{S}} \Pr(s_{i+1,k}|s_{ij}) \Pr(r_{i+1}|s_{i+1,k}, g_{i+1}) \beta_{i+1,k} \quad (30)$$

We will call this decoding algorithm the “iterative state estimation” algorithm and it is illustrated in figure 5. This algorithm can now be used over bit-flipping channels unlike the one-off state estimation algorithm. The computation order used in the following experiments was that the channel nodes were first updated as a chain. Next the symbol nodes were initialised in the traditional belief propagation manner. Then iteration over the graph was started. The nodes were repeatedly updated in the following order: the check nodes, the symbol nodes, the channel nodes and then the symbol nodes.

A further improvement would be if one had additional information of the true channel state for each bit received – the traditional LDPC decoding algorithm can then be used with input probabilities derived using the state information. We will call this the “genie-aided” algorithm.

5 Performance on Markov channel

An $R = 1/2$, $N = 8000$ regular low-density parity-check code with column weight 3 created by David J.C. MacKay¹ was used to test the above algorithms.

5.1 Bursts of total data loss

A channel with bit flipping noise was first studied. Two states were used: one with $p_f = 50\%$ and one with $p_f = 0\%$; the former had typical burst length of 10 and the latter had a variable typical burst length, figure 6(a). The results of experimental simulations with three different decoding algorithms and the corresponding Shannon limits are shown in figure 6(b).

¹8000.4000.3.483

available

from

<http://wol.ra.phy.cam.ac.uk/mackay/codes/data.html>

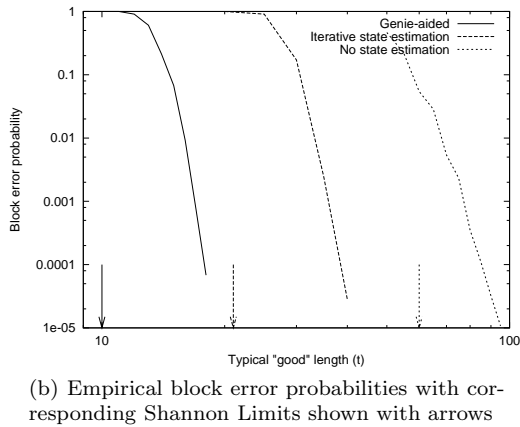
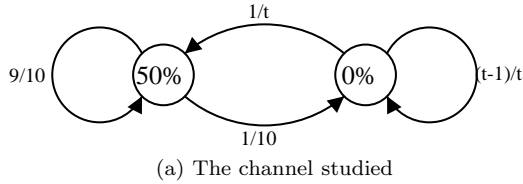


Figure 6: Bit-flipping channel with variable length in the “good” state

5.2 Gaussian noise

A channel with gaussian noise was also studied. Two states were used with a separation in noise level of 7dB (to match [13]). The typical run-length in each state was the same – values of 3, 10 and 30 were studied. The average SNR was varied and the empirical block error rate was measured using all four algorithms described above. The results are shown in figure 7.

5.3 Bit-flipping noise

The same experiment was then repeated using bit-flipping noise. To make the experiment more easily comparable to the previous experiment this noise was assumed to come from taking a hard decision from gaussian noise – thus the two states were chosen so as to have a 7dB difference in noise level. One would expect a 2dB drop from taking hard decisions [5]. Empirical results are shown in figure 8.

The capacity of this type of channel is shown in figure 9.

5.4 Typicality of the channel

It can be seen that as the burst length increases the performance curves become flatter. It is thought that this is due to the variation of the total noise energy of the channel, over the studied block size of 8000, increasing as the typical burst length increases. For a burst length of 30 an estimate of the probability density function of the number of bits transmitted in the noisy state is shown in figure 10(a). The standard deviation of this distribution against typical burst length is shown in figure 10(b).

5.5 Iterations

For the latter bit-flipping experiment (section 5.3) the evolution of $\Pr(s_{ij} | \mathbf{r}, \{g_k : k \neq i\})$ was studied. For the first

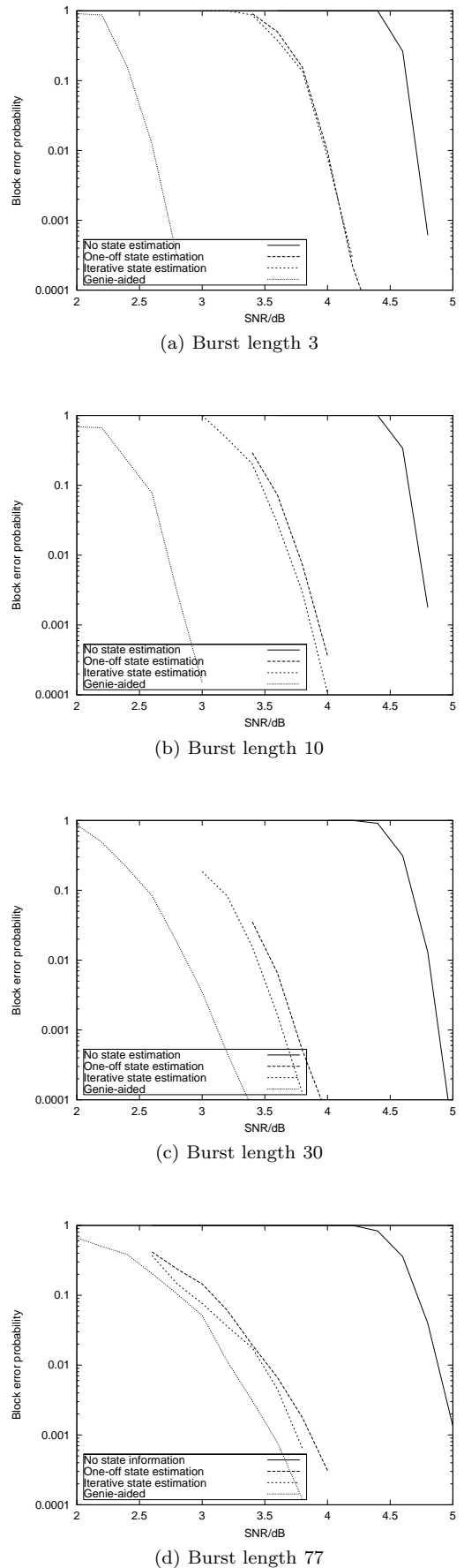
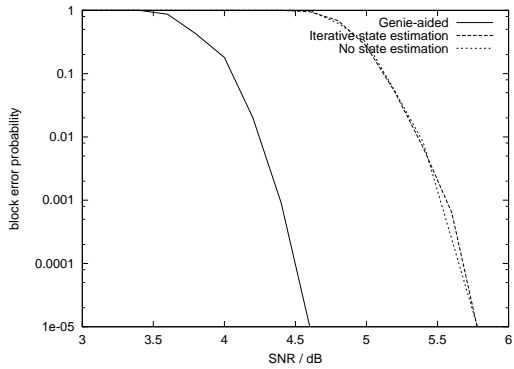
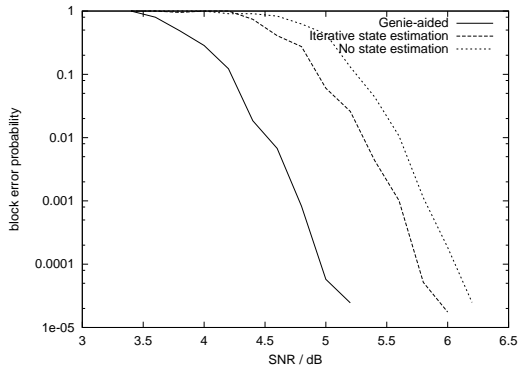


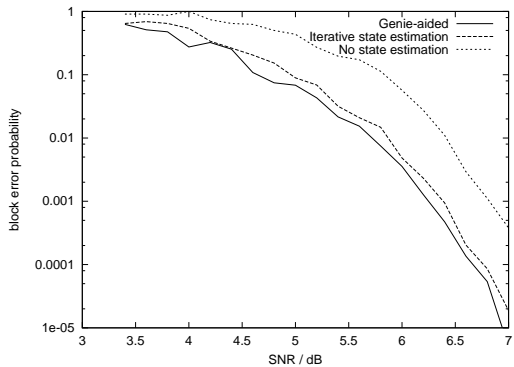
Figure 7: Gaussian noise with 7dB separation between two states with equal typical run-lengths



(a) Burst length 3



(b) Burst length 30



(c) Burst length 300

Figure 8: Bit-flipping noise with 7dB separation between two states with equal typical run-lengths

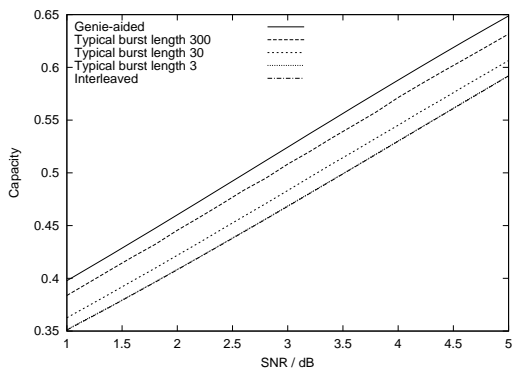


Figure 9: Capacity of two-state bit-flipping Markov channels with 7dB noise separation – note the interleaved and the typical burst length of 3 lines fall on top of each other at this scale

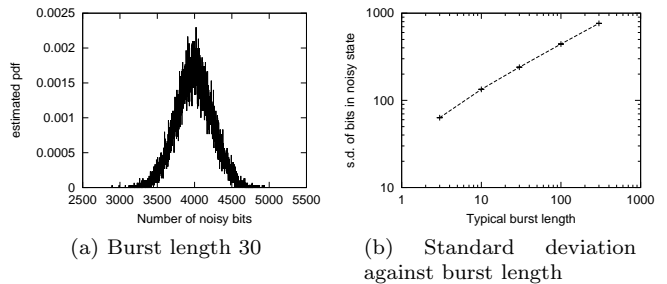


Figure 10: Distribution of number transmissions in the noisy state for an $N = 8000$ block over a 2-state Markov channel

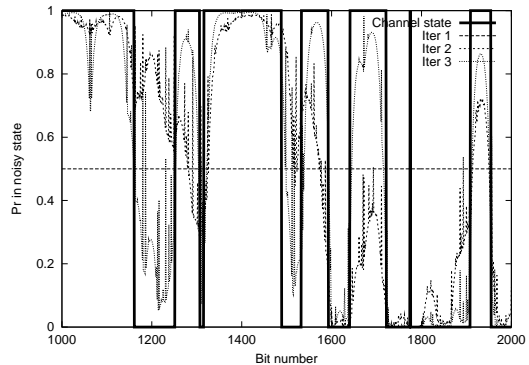


Figure 11: Evolution of the inference of state ($\Pr(s_{ij}|\mathbf{r}, \{g_k : k \neq i\})$) for a two-state bit-flipping channel with average noise of 4.8dB, separation of 7dB and typical burst length of 100.

iteration one starts with a 50:50 distribution and it is expected to tend towards the true state distribution. An example of the evolution is shown in figure 11. Correct decoding was achieved in three iterations for the example shown. The state information tends pretty quickly towards the true value – large errors in iteration 2 can be seen around bit numbers 1200 and 1700 but these are corrected by the next iteration.

The number of iterations before the iterative state estimation algorithm reached a decoding was also recorded for the gaussian noise experiment (section 5.2) and is shown in figure 12. It can be seen that fewer iterations are needed for a successful decoding as the typical burst length increases. This suggests that decoding is easier as the typical burst length increases. The most difficult region for the channel nodes to infer the true state is near a state transition. As the typical burst length increases these regions become less significant and hence the decoding process can be expected to be easier.

6 Discussion

For bit-flipping noise the iterative state estimation algorithm should be compared with the Gilbert-Elliott capacity as this represents the best possible performance over the bit-flipping channel with no channel state information. The first experiment (section 5.1) shows that the iterative state estimation algorithm, despite not reaching this capacity with the chosen code, is producing the expected gain over the no state information case. The further experi-

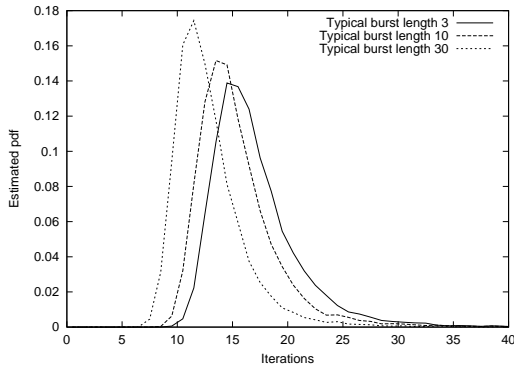


Figure 12: Empirical pdf of number of iterations until the iterative state estimation algorithm terminates. Gaussian noise was used with an SNR difference of 7dB and the average SNR was chosen so the block error probability was 10^{-3} .

ments on the bit-flipping channel (section 5.3) also show that the iterative state estimation algorithm is producing similarly good gains but we can further see the codes are at least 2dB from capacity at a block error probability of 10^{-3} .

The performance of the algorithms over the gaussian noise channel is hard to assess as the channel capacity has not been calculated to the author's knowledge. It can be seen that behaviour similar to the bit-flipping case is observed – the code performance increases as the typical burst length increases. However this increase happens faster than for the bit-flipping case, probably due to the extra information in the received amplitudes. It is interesting to compare the one-off state estimation algorithm and the iterative state estimation algorithm. For very short bursts iterative state estimation can not offer any significant advantage over one-off state estimation. When the typical burst length gets up to 30 the gain is only 0.1dB.

Earlier work on a block bursty channel [13] with a hop length of 10 bits and a code of the same rate suggested that larger gains might be found between one-off state estimation and iterative state estimation. A comparison can be done between the entropy per symbol of the state sequences of the two channels:

$$H(\text{block bursty channel}) = \frac{1}{\text{hop length}} \quad (31)$$

$$H(\text{our Markov channel}) = H_2\left(\frac{1}{\text{burst length}}\right) \quad (32)$$

For a hop length of 10 one needs a typical burst length of 77 for these two expressions to be equal. However the experiments with a burst length of 77 suggest that a gain of about 0.1dB (rather than 0.2dB for the block bursty channel as reported in [13]) is being seen.

Many iteration schemes could be used on the belief propagation graph. It would be worth investigating other schemes and comparing their expected computational complexity. Quicker decodings may be achieved in some cases by, for example, only evaluating the channel nodes after a few iterations over just the symbol and check nodes.

No attempt was made to optimize the code used. It is hoped that a larger block-size irregular code could perform better as seen with the AWGN channel [8]. Confirmation of this would be useful further work. Further criteria could also be used – the removal of “near” codewords (codewords

that lead to low weight syndromes when a set of bits transmitted at close times are corrupted) could be advantageous as a burst of noise is likely to corrupt several nearby bits.

7 Conclusion

For the bit-flipping noise Markov channel we have shown that extending the LDPC belief propagation decoding algorithm to include channel estimation leads to a gain of the same order as the difference between the interleaved capacity and the Gilbert-Elliott capacity. With gaussian noise the most significant gain comes from estimating the channel state once. As the typical burst length increases iterative state estimation produces an increasing gain over one-off state estimation.

References

- [1] Lalit R. Bahl, John Cocke, Frederick Jelinek, and Josef Raviv. Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Transactions on Information Theory*, 20(2):284–287, March 1974.
- [2] R. Durbin et al. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge University Press, 1998.
- [3] Javier Garcia-Frias. Decoding of low-density parity check codes over finite-state binary Markov channels. In *IEEE International Symposium on Information Theory*, 2001.
- [4] Andrea J. Goldsmith and Pravin P. Varaiya. Capacity, mutual information, and coding for finite-state Markov channels. *IEEE Transactions on Information Theory*, 42(3):868–886, May 1996.
- [5] Rolf Johannesson and Kamil Sh. Zigangirov. *Fundamentals of Convolutional Coding*. IEEE, 1998.
- [6] David J.C. MacKay. Good error correcting codes based on very sparse matrices. *IEEE Transactions on Information Theory*, 45(2):399–431, 1999.
- [7] Mordechai Mushkin and Israel Bar-David. Capacity and coding for the Gilbert-Elliott channels. *IEEE Transactions on Information Theory*, 35(6):1277–1290, November 1989.
- [8] Thomas J. Richardson, M. Amin Shokrollahi, and Rüdiger L. Urbanke. Design of capacity-approaching irregular low-density parity-check codes. *IEEE Transactions on Information Theory*, 47(2):619–637, February 2001.
- [9] Peter Sweeney. Discussion at the International Symposium on Communication Theory and Applications, July 2001.
- [10] Tadashi Wadayama. An iterative decoding algorithm of low density parity check codes for hidden Markov noise channels. In *International Symposium on Information Theory and Its Applications*, November 2000.
- [11] Hong Shen Wang and Nader Moayeri. Finite-state Markov channel – a useful model for radio communication channels. *IEEE Transactions on Vehicular Technology*, 44(1):163–171, February 1995.
- [12] Andrew P. Worthen. *Codes and iterative receivers for wireless communication systems*. PhD thesis, University of Michigan, 2001. Available on the web at <http://www.eecs.umich.edu/~worthena/>.
- [13] Andrew P. Worthen and Wayne E. Stark. Low-density parity check codes for fading channels with memory. In *Proceedings of the 36th Annual Allerton Conference on Communication, Control and Computing*, 1998.
- [14] Andrew P. Worthen and Wayne E. Stark. Unified design of iterative receivers using factor graphs. *IEEE Transactions on Information Theory*, 47(2):843–849, February 2001.
- [15] Michele Zorzi, Ramesh R. Rao, and Laurence B. Milstein. On the accuracy of a first-order Markov model for data transmission on fading channels. In *ICUPC'95, Tokyo, Japan*, November 1995.