MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Ripple Design of LT Codes for AWGN Channel

Sorensen, J.H.; Koike-Akino, T.; Orlik, P.

TR2012-053 July 2012

Abstract

In this paper, we present an analytical framework for designing LT codes in additive white Gaussian noise (AWGN) channels. We show that some of analytical results from binary erasure channels (BEC) also hold in AWGN channels with slight modifications. This enables us to apply a ripple-based design approach, which until now has only been used in the BEC. LT codes designed by this way show promising performance which is near the Shannon limit even with short codewords.

IEEE Intternational Symposium on Information Theory Proceedings (ISIT)

Ripple Design of LT Codes for AWGN Channel

Jesper H. Sørensen*†, Toshiaki Koike-Akino†, Philip Orlik†, Jan Østergaard*, and Petar Popovski*
*Aalborg University, Department of Electronic Systems, Fredrik Bajers Vej 7, 9220 Aalborg, Denmark
†Mitsubishi Electric Research Laboratories (MERL), 201 Broadway, Cambridge, MA 02139, USA
E-mail: {jhs, jo, petarp}@es.aau.dk, {koike, porlik}@merl.com

***Abstract*—In this paper, we present an analytical framework for designing LT codes in additive white Gaussian noise (AWGN) channels. We show that some of analytical results from binary erasure channels (BEC) also hold in AWGN channels with slight modifications. This enables us to apply a ripple-based design approach, which until now has only been used in the BEC. LT codes designed by this way show promising performance which is near the Shannon limit even with short codewords.**

I. Introduction

LT codes [1] were the first practical examples of a rateless erasure correcting code which approaches capacity for increasing message length. Raptor codes [2] were later developed as an extension of LT codes. Such rateless codes may potentially generate an infinite amount of encoded symbols from finite input symbols. An important element in the design of both LT and Raptor codes for the binary erasure channel (BEC) is a parameter called the *ripple*. The performance depends significantly on how this parameter evolves during decoding, and thus successful designs have mostly focused on this. Although LT codes were originally developed for the BEC, several works have recently focused on designing such codes for noisy channels.

Design of Raptor codes for the binary symmetric channel (BSC) and binary-input additive white Gaussian noise (Bi-AWGN) channel is treated in [3]. This design is based on the Gaussian approximation method [4], which is used to derive constraints for the degree distribution of the LT code. In [5, 6], approaches based on EXIT charts are applied to the design of LT and Raptor codes, respectively. Another design of Raptor codes for arbitrary symmetric channels is presented in [7], where an analogue to the ripple is defined as the increase in correct bit estimates in a given decoding round. The design is based on finding an optimal value for this measure.

In this paper, we present an analytical framework for the design of LT codes in the AWGN channel. It has strong analogies to the framework presented in [1] for the BEC. Interestingly, we show that key analytical results in the BEC also hold even in the AWGN channel. This enables us to make a ripple-based design for the AWGN channel with slight modifications, which exploit characteristics unique in AWGN. The main contribution of this work is a bridge between the work in the BEC and the AWGN channel, and it can help further design extensions for such noisy channels.

II. Background of LT Codes

In this section, an overview of LT codes is presented. Assume we wish to transmit a given amount of data, which is divided into k *input symbols*. From these input symbols a potentially infinite amount of encoded symbols, called *output symbols*, are generated. Output symbols are exclusive-or (XOR) combinations of input symbols. The number of input symbols used in the XOR is called the *degree* of the output symbol, and all input symbols contained in an output symbol are called *neighbors* of the output symbol. The output symbols follow a certain degree distribution, $\Omega(d)$. The encoding process can be broken down into three steps: 1) Randomly choose a degree d by sampling $\Omega(d)$. 2) Choose uniformly at random d of the k input symbols. 3) Perform XOR of the d chosen input symbols. The resulting symbol is the output symbol. This process can be iterated as many times as needed, that results in a rateless code.

A. Decoding in AWGN Channel

A widely used decoder for LT codes is a belief propagation (BP) decoder. For the BP decoder, messages are passed between neighboring symbols, i.e. from output symbols to input symbols or *vice versa*. Such a message reflects the current belief of the sender on the value of an input symbol. The belief is quantified by the log-likelihood ratio (LLR), defined as $\ln\left(\frac{\Pr(X_i=1|Y)}{\Pr(X_i=0|Y)}\right)$, where X_i is the i-th binary input symbol and Y is the AWGN channel output symbols vector of potentially infinite length. The o-th output symbol is denoted as Y_o. The ℓ-th round of the BP decoder starts with all output symbols passing a message, $m_{o,i}^{\ell}$ to all their neighboring input symbols. Based on those messages, the input symbols pass a message, $m_{i,o}^{\ell}$, back to all their neighboring output symbols. These rounds continue until a specified stop criterion has been reached, e.g. a certain number of rounds or a target error rate.

The belief messages are updated as follows [3]:

$$m_{i,o}^{\ell} = \sum_{o' \neq o} m_{o',i}^{\ell-1},$$

$$m_{o,i}^{\ell} = 2 \operatorname{arctanh}\left(\tanh\left(\frac{Z_o}{2}\right) \cdot \prod_{i' \neq i} \tanh\left(\frac{m_{i',o}^{\ell}}{2}\right)\right), \quad (1)$$

where Z_o is the LLR of Y_o based only on the channel output. The product (sum) is taken over all neighboring input (output) symbols other than the message recipient i (o) itself.

B. Decoding in BEC

In the BEC, the BP decoder can be significantly simplified since all received symbols in decoding are completely reliable.

This implies that the decoder can perform the logical XOR operations inversely from the encoding process. First, all degree-1 output symbols are identified and moved to a storage referred to as the *ripple*. Symbols in the ripple are *processed* one by one, in which they are XOR'ed with all output symbols who have them as neighbors. Once a symbol has been processed, it is removed from the ripple and considered decoded. The processing of symbols in the ripple will potentially reduce some of the buffered symbols to degree one, in which case they are moved to the ripple. This is called a symbol *release*. The decoder can then process symbols in a successive fashion.

The ripple is an important parameter in the BEC. In [1], a trade-off was described. If a new symbol is released every time, there is a risk that it is already in the ripple, that causes redundancy. It suggests that the ripple size should be kept low. However, in order to decrease the risk of decoding failure, which occurs when the ripple size is zero, the ripple size should be kept high enough. A good solution to this trade-off is the main design problem in the BEC. In the following analysis, we use information-theoretic tools to present an analytical framework which generalizes the ripple-based approach towards the AWGN channel.

III. Ripple-Oriented Analysis

We consider the BP decoder described in section II-A, with a slight modification, in order to facilitate the analysis. Instead of letting all input symbols pass a new message in a round, we allow only one randomly chosen input symbol to do so. All other input symbols pass the message in the previous round. This modified BP decoder for input-to-output message updating is expressed as

$$m_{i,o}^{\ell} = \begin{cases} \sum_{o' \neq o} m_{o',i}^{\ell-1}, & \text{if } i \text{ is allowed to pass,} \\ m_{i,o}^{\ell-1}, & \text{if } i \text{ is not allowed to pass.} \end{cases} \tag{2}$$

This modification is known as a random scheduling for BP. It is known that such a sequential scheduling offers better convergence performance than a standard parallel scheduling. Note that a code designed by this analysis can be decoded by any BP decoder scheduling.

A. Analytical Framework

We first express the entropy, $H(Z_i^{\ell})$, of the i-th input symbol after ℓ decoding rounds as follows:

$$\begin{aligned} H(Z_i^{\ell}) &= -\sum_{x \in \{0,1\}} \Pr(X_i = x|Y) \log_2\big(\Pr(X_i = x|Y)\big), \\ \Pr(X_i = 1|Y) &= \frac{\exp(Z_i^{\ell})}{1 + \exp(Z_i^{\ell})}, \\ \Pr(X_i = 0|Y) &= 1 - \Pr(X_i = 1|Y), \\ Z_i^{\ell} &= \sum_o m_{o,i}^{\ell}. \end{aligned} \tag{3}$$

When an input symbol passes a new message to its neighbors, we refer to the information it holds as *processed information*. After the ℓ-th decoding round, the processed information, I_P^{ℓ}, is defined as the total amount of information passed from input symbols to output symbols. It is given as

$$I_P^{\ell} = \sum_{i=1}^{k} \big(1 - H(Z_i^p)\big), \qquad Z_i^p = \frac{\sum_o m_{i,o}^{\ell}}{d-1}, \tag{4}$$

where $H(Z_i^p)$ is interpreted as the entropy of the i-th input symbol at the point of its last message passing. Directly following from (4), we have the definition of *unprocessed information*, $I_L^{\ell} = \sum_{i=1}^{k} H(Z_i^p)$.

When deciding which input symbol should be allowed to pass a new message, a uniform random selection is performed among the input symbols, which hold information not yet passed to its neighbors. We say that these candidates contribute to the *information ripple*. The information ripple, I_R^{ℓ}, after the ℓ-th decoding round, is defined as the total amount of information, held by the input symbols which have not yet been passed to the output symbols. We have

$$I_R^{\ell} = \sum_{i=1}^{k} \big(H(Z_i^p) - H(Z_i^{\ell})\big). \tag{5}$$

After the input symbol has passed its message, the output symbols obtain a chance to pass messages back to their neighboring input symbols. Only output symbols, which are neighbors to the last message passing input symbol, have new information to pass. This new information is referred to as *released information*, denoted I_Q^{ℓ}. It is expressed as

$$\begin{aligned} I_Q^{\ell} &= \sum_{i=1}^{k} \big(H(Z_i^p) - H(Z_i^{p+})\big), \\ Z_i^{p+} &= Z_i^p + \sum_o \big(m_{o,i}^{\ell} - m_{o,i}^{\ell-1}\big), \end{aligned} \tag{6}$$

where $H(Z_i^{p+})$ is interpreted as the entropy of the i-th input symbol when processed and newly released information is taken into account.

Here, I_Q^{ℓ} is defined by $H(Z_i^p)$ as reference, which is the information known by the output symbols. Hence, I_Q^{ℓ} can be regarded as the amount of new information passed to the input symbols, as seen from an output symbol perspective. In fact, this is not the true amount of new information since it might be combined with information in the ripple. For this case, the actual reference is $H(Z_i^{\ell-1})$ and we can define the actual amount of information added to the ripple, I_A^{ℓ}, as follows:

$$\begin{aligned} I_A^{\ell} &= \sum_{i=1}^{k} \left(H(Z_i^{\ell-1}) - H\left(Z_i^{\ell-1} + \sum_o \big(m_{o,i}^{\ell} - m_{o,i}^{\ell-1}\big) \right) \right) \\ &= \sum_{i=1}^{k} \Big(H\left(Z_i^{\ell-1}\right) - H\left(Z_i^{\ell}\right) \Big). \end{aligned} \tag{7}$$

The quantities defined in (3) through (7) are illustrated in Fig. 1, where the entropy of a single input symbol has been plotted as a function of its LLR. Due to the convexity of the entropy function (except at very low LLR), we have $I_A^{\ell} < I_Q^{\ell}$, which means loss of information. This is analogous to the risk of redundancy for nonzero ripple in the BEC.

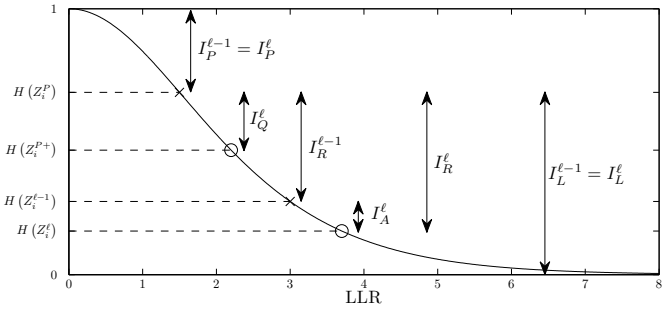

Fig. 1. Entropy as a function of LLR for an input symbol i. The contribution to the defined quantities is annotated, assuming that the symbol has not passed a new message, but received new information from its neighbors.

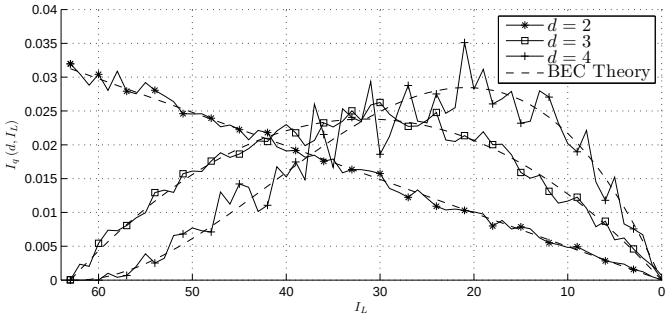

Fig. 2. Comparison of simulated I_Q as a function of d and quantized I_L and corresponding curves from (8).

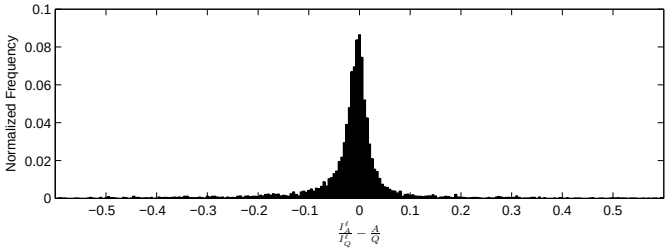

Fig. 3. Histogram of the difference between simulated I_A^{ℓ}/I_Q^{ℓ} from AWGN channel and theoretic $A(L)/Q(L)$ from the BEC.

B. First Moment of Information Ripple

In general, there is a strong relation between the quantities in (3) through (7) and the terms defined in section II-B for the BEC decoder. They are essentially continuous entropy counterparts of the discrete symbol based versions from the BEC. One interesting quantity in a ripple-based design is the expected amount of released information from an output symbol of degree d as a function of the amount of unprocessed information. It expresses the universal connection between the degree distribution, which is the design parameter, and the rate of recovery of new information during the decoding process. It was derived in [1] for the BEC, more specifically

$$\begin{aligned} &q(1,k) = 1, \\ &q(d,L) = \frac{d(d-1)L\prod_{j=0}^{d-3}(k-(L+1)-j)}{\prod_{j=0}^{d-1}(k-j)}, \\ &\qquad \text{for } d = 2,...,k, \text{ and } L = k-d+1,...,1, \\ &q(d,L) = 0, \text{ for all other } d \text{ and } L, \end{aligned} \tag{8}$$

where L is the amount of unprocessed input symbols. Deriving the AWGN channel equivalent to (8) is outside the scope of this paper. However, we can easily obtain an understanding of its behavior by simulating an LT code in an AWGN channel and logging I_Q^{ℓ} versus I_L^{ℓ} for different degrees. In order to compare with (8), we quantize this data to integer values of I_L^{ℓ} and normalize such that it sums up to one. We thus have the fraction of the released information, which is released when I_L bits remain unprocessed. This is denoted as $I_q(d, I_L)$ for output symbols of degree d and is defined as

$$I_q(d, I_L) = \frac{\sum_{\ell:|I_L^{\ell}-I_L|<0.5} I_Q^{\ell}}{\sum_{\ell} I_Q^{\ell}}, \tag{9}$$

where only information from output symbols of degree d is included. Fig. 2 shows a plot of the results at a signal-to-noise power ratio (SNR) of 5 dB. It reveals a clear correspondence between the theory derived for the BEC and what is observed in the AWGN channel.

However, as described in connection with (7), not all released information is added to the ripple. In order to determine how much, we must know $\frac{I_A^{\ell}}{I_Q^{\ell}}$, the ratio of released information which is added to the ripple. The BEC counterpart is $\frac{A(L)}{Q(L)} = \frac{L-R(L)}{L}$, where $A(L)$, $Q(L)$ and $R(L)$ are symbols added to the ripple, released symbols and symbols in the ripple after the $(k-L)$-th decoding round, respectively.

For the AWGN channel, $\frac{I_A^{\ell}}{I_Q^{\ell}}$ has been determined for $0 < R \leq 20$ and $0 < L \leq 64$ through a simulation similar to the one used for determining $I_q(d, I_L)$. Fig. 3 shows a histogram of $\frac{I_A^{\ell}}{I_Q^{\ell}} - \frac{A(L)}{Q(L)}$, which illustrates that the AWGN channel behaves as the BEC with a small perturbation.

Based on $q(d, L)$ and the fact that $\frac{A(L)}{Q(L)} = \frac{L-R(L)}{L}$ holds in the BEC, it is possible to control the expected amount of symbols added to the ripple in each decoding round, through the choice of degree distribution. We can now conclude that this theory also holds in the AWGN channel. Hence, the same ripple-based design approach can be used for the AWGN channel, in order to control the first moment of the ripple. This suggests that degree distributions designed for the BEC should also perform well in the AWGN channel, which was also concluded in [8]. However, it was also mentioned that there is still a room for improvement, and thus behavioral differences between the BEC and the AWGN channel. These differences should be evident in the higher moments of the ripple. We take this into account in our design, which is described next.

IV. Ripple-Based Design in AWGN

A typical approach to design LT codes is to choose design criterion for the first moment of the ripple and meet it through proper choice of the degree distribution. The first moment criterion is chosen based on heuristic assumptions about the second moment behavior. This approach was taken in both

[1] and [2] for the BEC. Using the analytical framework presented in the previous section, we will apply the ripple-based approach to designing LT codes in the AWGN channel.

In [2], the assumptions about the second moment behavior of the ripple in the BEC was based on a random walk model. It was assumed that the ripple size either increases or decreases by one, with equal probabilities, in each decoding round, i.e., $R^{\ell+1} = R^{\ell} \pm 1$. Since one symbol is processed in each round in the BEC, L rounds will remain, and we can calculate the expected distance from the origin, $\mathbb{E}\big[|R^{\ell+L} - R^{\ell}|\big]$ ($\mathbb{E}[\cdot]$ denotes an expectation), after the random walk as follows:

$$\mathbb{E}\big[\big(R^{\ell+L} - R^{\ell}\big)^2\big] = \sum_{j=\ell}^{\ell+L-1} \mathbb{E}\big[\big(R^{j+1} - R^{j}\big)^2\big] = L,$$
$$\mathbb{E}\big[|R^{\ell+L} - R^{\ell}|\big] = \sqrt{L}. \tag{10}$$

Based on this it was argued that the expected ripple should be kept at $c\sqrt{L}$, for a properly chosen constant c, when L symbols remain unprocessed, i.e. in the $(k-L)$-th decoding round. We will adopt this approach, for our choice of expected information ripple for the AWGN channel with a slight modification which deals with the characteristics of this channel.

First, we note that a decoding round does not necessarily result in the processing of 1 bit, i.e. $H(Z_i^p) - H(Z_i^{\ell}) \leq 1$. Hence, we assume that the ripple will increase or decrease by α with equal probabilities. Moreover, it is possible that $H(Z_i^{\ell}) > H(Z_i^p)$, i.e. the messages passed from output symbols to input symbol i, since last time it was allowed to pass, has increased the entropy of that input symbol. In this case, we actually have negative decoding progress. Numerical experiments show that this happens more frequently later in the decoding progress, which results in an increasing number of decoding rounds per one bit of processed information. We assume a linear increase, such that the number of steps per processed bit is $\theta = \beta(1 - \frac{I_L^{\ell}}{k}) + 1$. Hence, we obtain

$$\mathbb{E}\big[\big(R^{\ell+L} - R^{\ell}\big)^2\big] = \sum_{j=\ell}^{\ell+\theta I_L^{\ell}-1} \mathbb{E}\big[\big(R^{j+1} - R^{j}\big)^2\big] = \alpha^2 \theta I_L^{\ell},$$
$$\mathbb{E}\big[|R^{\ell+L} - R^{\ell}|\big] = \alpha\sqrt{\beta(1 - \frac{I_L^{\ell}}{k}) + 1}\sqrt{I_L^{\ell}}. \tag{11}$$

It determines a desired information ripple evolution with properly chosen constants, α and β. Note that this model is based on heuristics, as in [1] and [2]. It has been confirmed that a higher-order polynomial model of θ had almost no gains.

The AWGN channel differs from the BEC in another significant way in BP decoding. Messages passed from output symbols may be misleading, in the sense that they contribute with an LLR of an opposite sign of the true value of the bit. If one or more bits are in error, more output symbols will need to be collected. In this case, the errors may propagate and make future output symbols misleading as well.

Consider the example where 2 out of k input symbols are in error and a new output symbol of degree 3 is received. The three neighbors of the new symbol are the two erroneous input symbols and one correct input symbol. When this new output symbol passes a message to an input symbol, it is based on a product of the messages from the two other neighbors. Hence, for the erroneous input symbols, yet another misleading message will be passed, while for the correct input symbol, the errors cancel out. If only one erroneous input symbol had been a neighbor, the output symbol would have made a helpful contribution. Similar examples can be made with higher numbers of errors. In general, we can say that if it is more likely that an even number of erroneous input symbols are neighbors to a new output symbol, compared to an odd number, then the errors will be self-perpetuating.

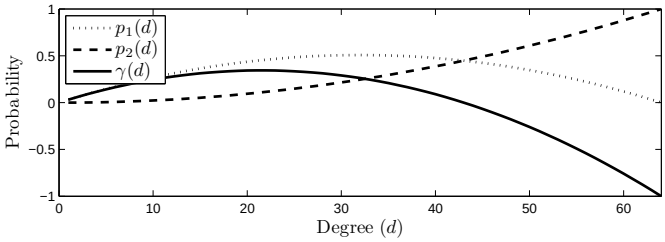

Fig. 4. Probabilities of having 1 and 2 erroneous neighbors, and the difference between them, as a function of degree.

This observation can be used to create a design criterion, where we focus on the cases of 1 and 2 erroneous neighbors, since these are most likely. We define $\gamma(d) = p_1(d) - p_2(d)$, where $p_e(d)$ is the probability that a new output symbol of degree d has e and only e erroneous neighbor(s) for $e = 1, 2$. They are expressed as $p_1(d) = \frac{\binom{d}{1}\binom{k-d}{1}}{\binom{k}{2}}$ and $p_2(d) = \frac{\binom{d}{2}\binom{k-d}{0}}{\binom{k}{2}}$ and plotted as a function of d in Fig. 4. The plots illustrate that high degree symbols should be avoided, whereas degrees in the lower half of the spectrum are more useful. In our design, we seek to maximize $\mathbb{E}[\gamma] = \sum_d \gamma(d)\Omega(d)$, under the constraints given by our choice of ripple.

We now summarize the analysis and the ripple-based design method for noisy channels. In section III-B, we showed that (8), which was originally derived for the BEC, also holds in the AWGN channel. Moreover, we showed that $\frac{I_A^{\ell}}{I_Q^{\ell}} = \frac{I_L^{\ell} - I_R^{\ell}}{I_L^{\ell}}$ is also analogous to the BEC. If we quantize the decoding process to integer values of I_L, we can express the expected evolution of the information ripple for each processed bit:

$$I_R(k) = \alpha\sqrt{I_L},$$
$$I_R(I_L - 1) = I_R(I_L) - 1 + \frac{I_L - I_R(I_L)}{I_L}\sum_d I_Q(d, I_L),$$
$$\text{for } k > I_L \geq 0,$$
$$\sum_d I_Q(d, I_L) = \sum_d nI(X;Y)\Omega(d)I_q(d, I_L), \tag{12}$$

where $I(X;Y)$ is the mutual information of the AWGN channel and n is the number of symbols collected for decoding.

Combining our ripple constraint from (11) with (12), we

obtain the system of equations as follows:

$$\begin{bmatrix} q(1,k) & 0 & 0 \\ \vdots & \ddots & 0 \\ q(1,1) & \cdots & q(k,1) \\ \omega\gamma(1) & \cdots & \omega\gamma(k) \end{bmatrix} \begin{bmatrix} n'\Omega(1) \\ \vdots \\ n'\Omega(k) \end{bmatrix} = \begin{bmatrix} I_R(k) \\ \Delta I_R(k-1) \\ \vdots \\ \Delta I_R(1) \\ n'\omega\mathbb{E}[\gamma] \end{bmatrix}, \quad (13)$$

where $n' = nI(X;Y)$ and $\Delta I_R(I_L - 1) = \frac{(I_R(I_L-1)-I_R(I_L)+1)I_L}{I_L-I_R(I_L)}$. Here we have also added $\mathbb{E}[\gamma] = \sum_d \gamma(d)\Omega(d)$, which we wish to maximize. This equation has a multiplier $\omega \gg 1$ since this is a firm constraint. Note that n' acts as a normalization factor, which ensures a valid degree distribution.

We can then formalize the design problem as follows:

$$\max \mathbb{E}[\gamma] \text{ s.t. } |An'\Omega(d) - b|^2 < t, \quad (14)$$

where A is the matrix in (13), b is the vector on the RHS of (13) and t is an appropriately chosen tolerance level of the deviation from the target information ripple.

V. Numerical Results

Simulations have been performed in order to compare the proposed design, $\Omega(d)$, with existing degree distributions. A tolerance level of $t = 0.1$ has been chosen for the optimization problem in (14). A numerical optimization of the parameters α and β has been performed, which revealed that parameter pairs where $1.0 \leq \alpha \leq 1.4$ and $0 \leq \beta \leq 3$ perform well, depending on the SNR. In general α should increase for increasing SNR, while β should decrease. References in the first simulation are the Robust Soliton Distribution (RSD), with parameters $c = 0.1$ and $\delta = 1$, and a degree distribution designed by the proposed approach presented in this work, but with the ripple target in (10), which was suggested in [2]. We refer to this distribution as $\Gamma(d)$. A numerical optimization resulted in $c = 1.3$, which has been used. All distributions have been evaluated at $k = 64$ for an SNR of $0, 2, ..., 10$ dB using BPSK modulations. They are compared with respect to average overhead necessary in order to successfully decode all input symbols. An input symbol is considered successfully decoded if its error probability is below 10^{-12}. Fig. 5 shows the results, where it is evident that the approach presented in this work significantly outperforms the distributions designed for the BEC. Especially at low SNR, where the difference between the AWGN channel and the BEC is more significant, the proposed design excels. In this figure, we also present the lower bound of the overhead calculated by the Shannon limit in AWGN (for BPSK-constrained codebooks with an infinite length). As we can see, the proposed degree distribution approaches the Shannon limit in the low SNR regimes within 1 dB even for such a very short message length of $k = 64$.

Another simulation has been performed, where the purpose is to evaluate the degree distributions with respect to block error rate at specific overheads. A block error occurs when at least one symbol has an error probability above 10^{-12} after 2000 decoding rounds on $(1 + \epsilon)k$ output symbols. The simulation is performed at an SNR of 0 dB using the same parameters as in the first simulation. The results are shown in Fig. 6 and it is seen that the proposed design outperforms the other degree distributions at any overhead.

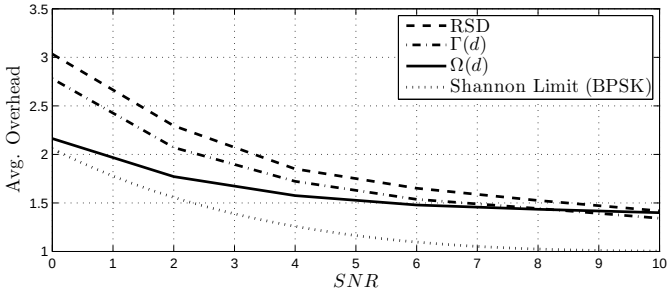

Fig. 5. Average overhead versus SNR for the simulated degree distributions.

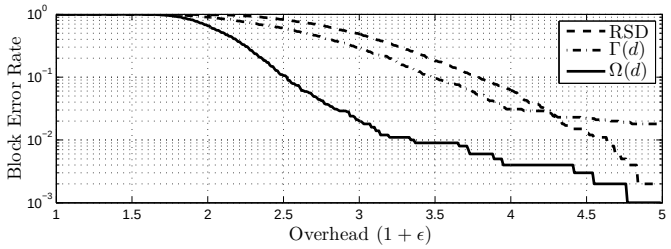

Fig. 6. Block error probability versus overhead for the simulated degree distributions.

VI. Conclusions

We have presented an analytical framework for LT codes in AWGN channels. Surprisingly, key analytical results known in the BEC can be applied to AWGN channels within this framework. This enables us to introduce a ripple-based design scheme in AWGN channels with a few modifications. LT codes based on this design show promising results compared to standard designs. In particular, the designed codes can approach the capacity in the low SNR regimes even for a very short message length. Our analytical framework is a strong tool for the ripple-based design in any noisy channels.

References

[1] M. Luby, "LT codes," in *The 43rd Annual IEEE Symposium on Foundations of Computer Science*, pp. 271–280, 2002.

[2] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory*, vol. 52, pp. 2551–2567, June 2006.

[3] O. Etesami and A. Shokrollahi, "Raptor codes on binary memoryless symmetric channels," *IEEE Transactions on Information Theory*, vol. 52, pp. 2033–2051, May 2006.

[4] S.-Y. Chung, T. Richardson, and R. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Transactions on Information Theory*, vol. 47, pp. 657–670, Feb. 2001.

[5] Z. Cheng, J. Castura, and Y. Mao, "On the design of Raptor codes for binary-input Gaussian channels," *IEEE Transactions on Communications*, vol. 57, pp. 3269–3277, Nov. 2009.

[6] I. Hussain, M. Xiao, and L. Rasmussen, "LT coded MSK over AWGN channels," in *The 6th International Symposium on Turbo Codes and Iterative Information Processing (ISTC)*, pp. 289–293, Sept. 2010.

[7] P. Pakzad and A. Shokrollahi, "Design principles for Raptor codes," in *IEEE ITW '06 Punta del Este.*, pp. 165–169, March 2006.

[8] R. Palanki and J. Yedidia, "Rateless codes on noisy channels," in *Available at www.merl.com/papers/TR2003-124/*, 2004.