

# Optimization of LDPC Codes for Receiver Frontends

Gottfried Lechner, Jossy Sayir  
Telecommunications Research Center Vienna (ftw.)  
Donaucitystr. 1/3  
1220 Vienna, Austria  
{lechner|sayir}@ftw.at

Ingmar Land  
Department of Communication Technology  
Aalborg University, Fredrik Bajers Vej 7, A3  
9220 Aalborg, Denmark  
il@kom.aau.dk

**Abstract**— The degree distribution of low-density parity-check (LDPC) codes is optimized for systems that iterate over the receiver frontend, e.g., soft detector, demodulator, equalizer, etc., and the LDPC decoder. The overall extrinsic information transfer (EXIT) function of an iterative LDPC decoder is computed, based on the code’s own EXIT chart, under the Gaussian assumption. While the optimization of the variable node distribution is a nonlinear problem, the optimization of the check node distribution is shown to be a linear problem. This fact is exploited to design codes where both the variable and the check node distributions are optimized, resulting in more robust constructions. The technique presented requires only knowledge of the measured EXIT function of the receiver frontend.

## I. INTRODUCTION

Low-Density Parity-Check (LDPC) codes [1], [2] achieve rates close to channel capacity for a wide range of channels. The performance of these codes is optimized in function of the degree distributions of the bipartite graph associated with the LDPC code. This optimization can be performed exactly by using density evolution [3] (without assuming specific distributions for the messages) or approximately by using extrinsic information transfer (EXIT) charts [4] assuming that the messages have a Gaussian distribution. When using EXIT charts, the code design reduces to a curve fitting problem<sup>1</sup>.

If we aim to iterate over a receiver *frontend* (e.g. a soft demapper for a specific modulation format or an equalizer for a channel with intersymbol-interference) and an error correcting code, we have to optimize the overall EXIT function of the code to match the EXIT function of the receiver frontend. The overall extrinsic information transfer function of a code optimized for the Gaussian memoryless channel is a step function [5]. This is consistent with the curve fitting approach of EXIT charts, since the extrinsic information transfer function of a memoryless Gaussian channel is constant and therefore, the best fit is achieved using a step function. On the other hand, for most other channels, e.g., channels with memory or non-binary channels, the EXIT function of the channel is not constant. Therefore, the frontend of the receiver can make use of a-priori information.

<sup>1</sup>This is motivated by the area property of the EXIT chart [4] which is only correct for the binary erasure channel (BEC) but results in good codes also for other channels.

When designing LDPC codes, it is common practice to fix the degree of the check nodes (i.e. use check-regular codes) and optimize the variable node distribution. One approach for matching the LDPC code to the frontend is to combine the variable nodes with the frontend as shown in [6]. This requires an analytical expression for the EXIT function of the frontend, which is provided in [6] by taking a polynomial approximation of the measured EXIT function. We follow a different approach, by matching the overall EXIT function of the LDPC decoder to the frontend. This only requires the measured EXIT function of the frontend and eliminates the need for an analytical approximation.

Designing the degree distributions of an LDPC code to match a given frontend is a nontrivial task, since the optimization problem is nonlinear, and therefore, search algorithms have to be employed to find good codes, as shown in [7]. In this paper, we optimize both the variable and the check node distributions. Ideally, one would like to optimize both distributions jointly, but this is usually not feasible. Therefore, we optimize them alternately. It can be shown that optimization of the check node distribution for a fixed variable node distribution is a linear problem that can be solved efficiently using linear programming. By optimizing the check node degree distribution for a fixed variable node degree distribution, the search for a good variable node degree distribution is far less complex, since the EXIT function of the LDPC code is less sensitive to the choice of the variable node distribution.

We introduce the system model used in this work in Section II. In Section III, we derive the EXIT function of an LDPC code and show its dependency on the degree distributions. These results are used in Section IV to design the LDPC code to match a given target function. Finally, Section V applies the code design method to turbo-demapping as an example.

## II. SYSTEM MODEL

We consider a system as shown in Figure 1. The transmitter encodes a binary vector  $\mathbf{u}$  to a binary vector  $\mathbf{x}$ . This block is interleaved, mapped to a vector  $\mathbf{t}$  of complex transmit symbols, and transmitted over the channel. For the moment, we do not make any assumptions about the channel. The receiver processes the received vector  $\mathbf{y}$  in a frontend and iterates

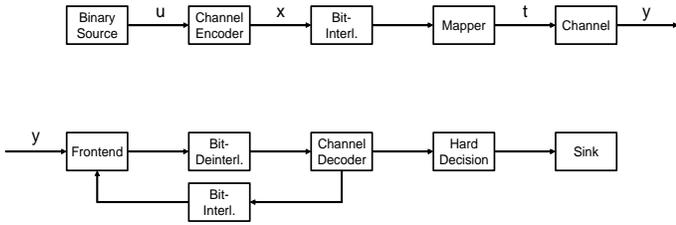


Fig. 1. System Model

between the (iterative) channel decoder and this frontend for a given number of iterations, before making a hard decision on the information vector. In this double iterative system, we perform iterations inside the LDPC decoder until it reaches a steady state (i.e., until the decoder converges to a codeword, or gets stuck at a point where the EXIT functions of variable and check nodes intersect) before returning to the outer iteration between the LDPC decoder and the demapper.

In order to optimize iterative processing between the frontend and the decoder, we aim to match the overall EXIT function of the code to the EXIT function of the frontend. Since, in our general model, we allow the frontend to perform several possible tasks, we do not make any assumptions about the transfer function of this component and regard it as given (obtained either analytically or via simulation) denoted by

$$I_{E,frontend} = T(I_A). \quad (1)$$

### III. EXIT FUNCTION OF LDPC CODES

In this section, we first will derive the EXIT function of an LDPC decoder and show how this function depends on the variable node degree distribution  $\lambda$  and the check node degree distribution  $\rho$  as defined in [8]. Since we are iterating between the frontend of the receiver and the channel decoder, which is an iterative system in itself, we have to assume some scheduling. Our analysis relies on asymptotic long blocks resulting in a graph for which the probability of cycles tends to zero and therefore, we are free to choose an arbitrary scheduling. Our assumption is that the decoder continues iterating until it reaches a steady state, before performing an outer iteration between the decoder and the frontend.

An EXIT chart of the LDPC decoder is shown in Figure 2. Quantities from variable to check nodes are denoted with the subscript VC and quantities from check to variable nodes with CV. When we allow the decoder to perform as many iterations as needed to converge to a steady state, the decoder will always get stuck in the smallest intersection between the variable and check node transfer curves denoted with  $I_{VC}^*$  and  $I_{CV}^*$ .

In order to get simple expressions for  $I_{VC}^*$  and  $I_{CV}^*$ , we use the duality property [4], which is an approximation for

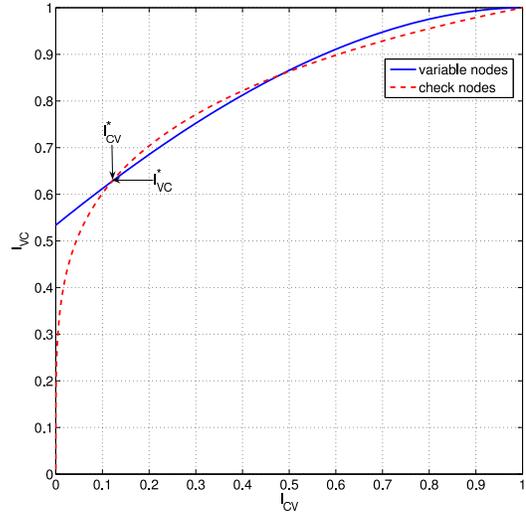


Fig. 2. EXIT chart of LDPC code

channels other than the BEC. Furthermore, we will use the function [4]

$$J(\sigma) = 1 - \int_{-\infty}^{\infty} \frac{e^{-\frac{(\theta-\sigma^2/2)^2}{2\sigma^2}}}{\sqrt{2\pi}\sigma} \log_2(1 + e^{-\theta}) d\theta. \quad (2)$$

Using this, we can write the variable and check node transfer functions as

$$I_{VC} = \sum_i \lambda_i \cdot J \left( \sqrt{(i-1)J^{-1}(I_{CV})^2 + J^{-1}(I_{A,LDPC})^2} \right), \quad (3)$$

$$I_{CV} = 1 - \sum_j \rho_j \cdot J \left( \sqrt{j-1} J^{-1}(1 - I_{VC}) \right). \quad (4)$$

The intersection point can thus be found by solving Equation 5, and this quantity is used to compute the extrinsic mutual information of the LDPC decoder as

$$I_{E,LDPC} = \sum_i \bar{\lambda}_i \cdot J \left( \sqrt{i} J^{-1}(I_{CV}^*) \right), \quad (6)$$

where  $\bar{\lambda}$  is the node perspective of the variable node degree distribution.

### IV. LDPC CODE DESIGN

Using Equations 5 and 6, we are now able to state our design problem. We aim to maximize the design rate  $R_d$  of

$$I_{CV}^* = 1 - \sum_j \rho_j \cdot J \left( \sqrt{j-1} J^{-1} \left( 1 - \sum_i \lambda_i \cdot J \left( \sqrt{(i-1)J^{-1}(I_{CV}^*)^2 + J^{-1}(I_{A,LDPC})^2} \right) \right) \right) \quad (5)$$

the code defined as

$$R_d = 1 - \frac{\sum \frac{\rho_i}{i}}{\sum \frac{\lambda_i}{i}} \quad (7)$$

under the constraint that the transfer function of the LDPC decoder is larger than a target function at every point

$$I_{E,LDPC}(I_A) \geq I_{E,target}(I_A). \quad (8)$$

This target function is given by the inverse transfer function of the frontend

$$I_{E,target}(I_A) = T^{-1}(I_A). \quad (9)$$

Looking at Equations 5, 6, and 7, one can observe that the maximization is nonlinear in  $\lambda$  given  $\rho$ , but linear in  $\rho$  given  $\lambda$ . Therefore, the optimization of  $\rho$  is much simpler than the optimization of  $\lambda$ . We will show in an example in Section V that the transfer function of the LDPC decoder is less sensitive to the choice of  $\lambda$  when  $\rho$  is optimized for each  $\lambda$ , than when an optimal  $\lambda$  is sought for a fixed  $\rho$ . Therefore, we can reduce the search space for  $\lambda$  significantly, which allows us to perform an exhaustive search.

#### Stability Condition

In order to design codes with no inherent error floor, the degree distributions have to satisfy the stability condition [3]. For the Gaussian channel with variance  $\sigma^2$ , the stability condition can be written as

$$\sum_i \rho_i \cdot (i - 1) < \frac{1}{\lambda_2} \frac{e^{2\sigma^2}}{\lambda_2}. \quad (10)$$

Note that the stability condition is a requirement for successful decoding of the LDPC code, that can be derived from properties of its EXIT chart in the top right corner. Therefore, in our double iterative setup, the stability condition is only relevant in the last overall iteration, when the LDPC decoder is expected to decode successfully, and its component EXIT curves do not intersect. If we assume that the messages between the frontend and the LDPC decoder are Gaussian distributed, the condition has thus to be satisfied for

$$\sigma = \frac{2}{J^{-1}(I_{E, Demapper, max})}, \quad (11)$$

where  $I_{E, Demapper, max}$  is the maximum of the EXIT function of the receiver frontend. This additional constraint is also linear in  $\rho$  and can therefore directly be included in the linear optimization problem.

#### V. EXAMPLE: TURBO DEMAPPING

As an example, we will apply the method described in the previous sections to iterative demapping and decoding (*turbo-demapping* [9]). Consider a system where the transmitter maps the encoded and interleaved data to a 16 QAM signal constellation. Depending on the applied mapping, the soft demapper in the receiver frontend has a certain information transfer function. For Gray mapping, this transfer function is approximately constant, meaning that turbo-demapping does

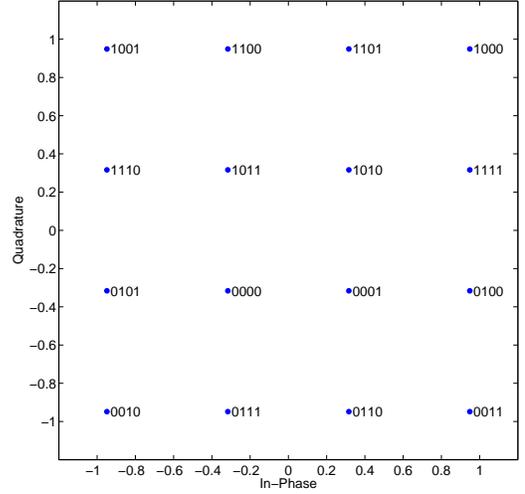


Fig. 3. 16 QAM constellation and set-partitioning mapping.

not lead to a significant performance improvement. However, for other mappings, like set-partitioning or non-unique symbol mappings [10], the transfer function of the demapper is not constant, which makes them a good candidate for turbo-demapping.

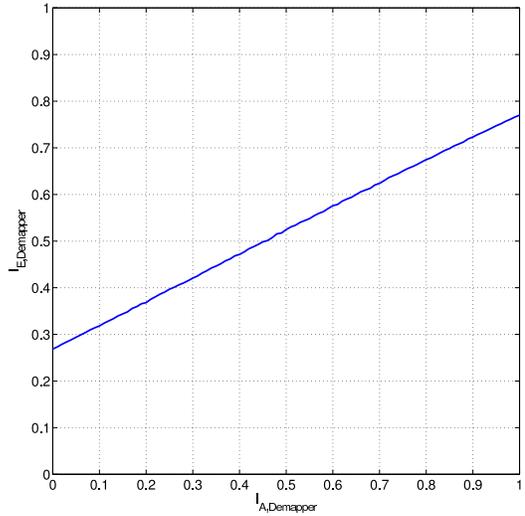


Fig. 4. EXIT function of the demapper

We will consider a 16 QAM signal constellation and set-partitioning mapping, transmitted over a memoryless additive white Gaussian noise (AWGN) channel with variance  $\sigma_{ch}^2$ . The constellation and the mapping are shown in Figure 3. We perform no constellation shaping, i.e. every constellation point is transmitted with the same probability. The transfer function of the soft demapper at  $\sigma_{ch}^2 = 0.28$  is shown in Figure 4.

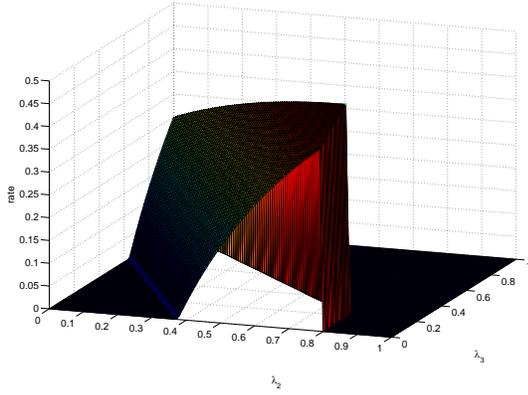


Fig. 5. Code search for fixed  $\rho$

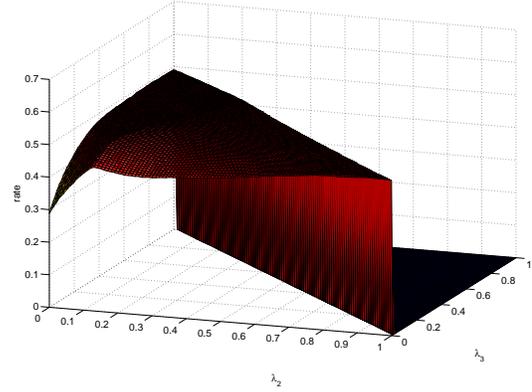


Fig. 6. Code search for optimized  $\rho$

Obviously, an LDPC decoder with a step-like transfer function can not achieve capacity for this system.

To demonstrate the advantage of optimizing both the variable and check node distribution, we compare two strategies. For both approaches, we restrict  $\lambda$  to have only three nonzero elements, namely for variable node degrees of two, three and ten. This restriction leads to only two degrees of freedom for the variable node degree distribution, making it easy to apply an exhaustive search over the complete search space.

First, the check node distribution is kept constant and regular with a check node degree of 4, which was found to deliver the highest rates, following the approach in [7]. The result of the exhaustive search is shown in Figure 5. It can be observed that the rate achieved is sensitive to the choice of  $\lambda$ . Furthermore, it can be seen that the fixed check degree distribution limits the maximum value of  $\lambda_2$ , due to the stability condition. We found a maximum rate of  $R_d = 0.46$ .

The second approach is to search for the best  $\lambda$  over the same search space, but to optimize  $\rho$  for every  $\lambda$ . The result of this search for a maximum check node degree of 30 is shown in Figure 6. Using the optimization of  $\rho$ , a maximum rate of  $R_d = 0.51$  was found, and the rate is less sensitive to the choice of  $\lambda$ . Even when setting all the variable nodes to degree two or three, the rates found were  $R_d = 0.48$  and  $R_d = 0.49$  respectively, still outperforming the best rate obtained using the first approach with a non-optimized check node degree distribution.

The EXIT function of the code with the optimized check node distribution matching the demapper function is shown in Figure 7 and the results are verified by bit error rate simulations with a codeword length of  $N = 10^5$ . For comparison, we also simulated the system with a rate 0.5 LDPC code optimized for an AWGN channel with BPSK mapping (having approximately a step function as transfer characteristic). From the results in Figure 8 it can be seen that the gain due to the code optimization is approximately 3.0dB.

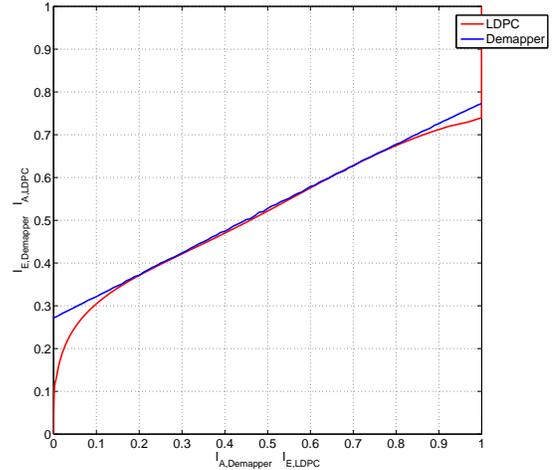


Fig. 7. EXIT chart of optimized LDPC code and demapper

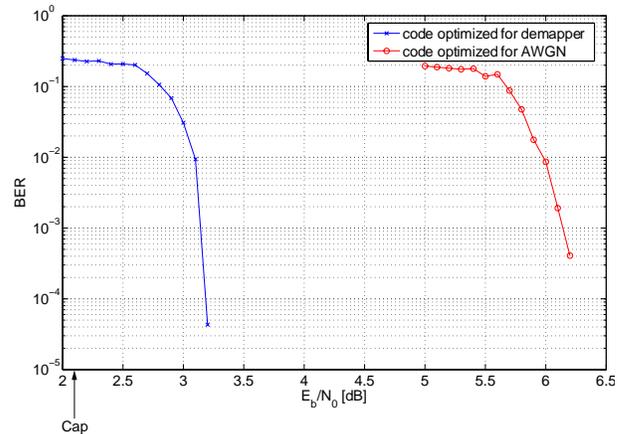


Fig. 8. Bit error rate simulations

## VI. CONCLUSION

We showed how the overall EXIT function of an LDPC decoder can be derived from its own EXIT chart using a Gaussian approximation. Optimizing this function, i.e. maximizing the rate of the code with constraints on its EXIT function, turned out to be a nonlinear problem for the optimization of the variable degree distribution, but is linear in the check node degree distribution. In contrast to conventional code design, where only the variable node distribution is optimized, we proposed to also optimize the check node distribution, resulting in a less complex code search and offering more degrees of freedom for the choice of the variable node distribution. This code optimization technique was applied to turbo-demapping as an example, and the results were verified by bit error rate simulations, demonstrating the importance of code optimization in iterative receivers.

## ACKNOWLEDGMENTS

This work was partly supported by the European network of excellence NEWCOM. ftw. is a research institution within the Austrian competence center funding program *Kplus*.

## REFERENCES

- [1] R.G. Gallager, *Low Density Parity Check Codes*, Number 21 in Research monograph series. MIT Press, Cambridge, Mass., 1963.
- [2] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Transactions on Information Theory*, vol. 45, no. 2, pp. 399–431, 1999.
- [3] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 47, pp. 619–637, Feb 2001.
- [4] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: model and erasure channel properties," *IEEE Transactions on Information Theory*, vol. 50, no. 11, pp. 2657–2673, 2004.
- [5] M. Peleg, A. Sanderovich, and S. Shamai, "On extrinsic information of good codes operating over discrete memoryless channels," available at <http://arxiv.org/pdf/cs.IT/0504028>.
- [6] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Transactions on Communications*, vol. 52, no. 4, pp. 670–678, April 2004.
- [7] A. Sanderovich, M. Peleg, and S. Shamai, "LDPC coded MIMO multiple access with iterative joint decoding," *IEEE Transactions on Information Theory*, vol. 51, no. 4, pp. 1437–1450, April 2005.
- [8] T. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Transactions on Information Theory*, vol. 47, pp. 599 – 618, 2001.
- [9] F. Schreckenbach and G. Bauch, "EXIT charts for iteratively decoded multilevel modulation," in *Proc. 12th European Signal Processing Conference (EUSIPCO)*, Vienna, Austria, September 2004, 2004.
- [10] F. Schreckenbach and P. Henkel, "Signal shaping using non-unique symbol mappings," in *Proc. 43rd Annual Allerton Conference on Communication, Control, and Computing*, Monticello, USA, September 2005.