Correcting Suboptimal Metrics in Iterative Decoders

Alex Alvarado, Víctor Núñez[§], Leszek Szczecinski[‡] and Erik Agrell
Department of Signals and Systems, Communication Systems Group
Chalmers University of Technology, Gothenburg, Sweden
[§]Department of Electronics Engineering
Universidad Técnica Federico Santa María, Valparaíso, Chile
[‡]INRS-EMT, Montreal, Canada
alex.alvarado@chalmers.se, {*nunez,leszek*}*@emt.inrs.ca*, *agrell@chalmers.se*,

***Abstract*—In this paper the issue of improving the performance of iterative decoders based on sub-optimal calculation of the messages exchanged during iterations (L-values) is addressed. It is well known in the literature that a simple—yet very effective—way to improve the performance of suboptimal iterative decoders is based on applying a scaling factor to the L-values. In this paper, starting with a theoretical model based on the so-called consistency condition of a random variable, we propose a methodology for correcting the L-values that relies only on the distribution of the soft information exchanged in the iterative process. This methodology gives a clear explanation of why the well-known linear scaling factor provides a very good performance. Additionally, the proposed methodology allows us to avoid the exhaustive search required otherwise. Numerical simulations show that for turbo codes the scaling factors found closely follow the optimum values, which translates to a close-to-optimal BER performance. Moreover, for LDPC codes, the proposed methodology produces a better BER performance compared with the known method in the literature.**

***Index Terms*—BER, Belief Propagation, Extrinsic Information, Turbo Codes, Iterative Decoding, L-values, LDPC codes.**

I. Introduction

Iterative decoding is an important element of modern communication systems due to the huge success of the so-called turbo codes (TC) discovered in 1993 [1], and the low-density parity check (LDPC) codes invented in 1962 [2] and rediscovered in 1999 [3]. In a very short time, both codes entered the commercial world and became part of the telecommunication standards (e.g., UMTS for TCs, and S-DVB for LDPC codes).

In this paper we aim to improve the iterative decoding algorithms based on sub-optimal calculation of the reliability metrics (L-values) exchanged during the iterations. The sub-optimality we consider is the result of the well-known max-log simplification considered in practice for its implementation simplicity. The improvement of the performance is sought via appropriate scaling of the L-values. Although this is a well known method, the search for the optimum scaling factor is conventionally based on a brute-force search, which requires extensive simulations. As opposed to this approach, we propose a method that has a low computational complexity, and provides a formal explanation about why a linear scaling factor works in practice. In this paper we show examples for TCs and LDPC codes, however, our method can be applied to any decoding algorithm, or to any other iterative process.

Both TCs and LDPC codes may be seen as instances of codes defined on the Tanner graphs [4], and may be decoded with a reasonably low complexity via message passing algorithms. Nodes of the graph generate messages that are propagated along edges to the directly connected nodes. Each message has a meaning of probability, and in the case of binary codes, messages are expressed in the logarithmic domain under the form of logarithmic likelihood ratios (L-values). Since the graphs contain cycles, an iterative process is generated. In TCs, the L-values are calculated for the set of nodes corresponding to one of the constituent codes, and they are propagated to another set of nodes which correspond to the second constituent code (assuming the most popular configuration where the TC is formed by two constituent codes). In LDPC codes, the L-values are calculated in each node and propagated to its immediate neighbors [3], [5], [6].

The calculation of the L-values at the nodes boils down to arithmetic operations carried out on the L-values obtained from the neighboring nodes. The optimal calculation requires a non-linear function in the form of $\log \sum_i \mathrm{e}^{L_i}$ where L_i are the input L-values. For a TC, optimal calculations produce the so-called maximum a posteriori probability (logAPP) implementing the BCJR algorithm [7]. For an LDPC code, the optimum algorithm is the so-called belief propagation (BP) algorithm, sometimes also referred as sum-product algorithm. However, for a hardware-efficient implementation, non-linear processing tends to be eliminated, which leads to the well-known max-log approximation $\log \sum_i \mathrm{e}^{L_i} \approx \max_i \{L_i\}$. Beside their simplicity, the resulting algorithms are insensitive to the linear scaling of the starting L-values (obtained from the channel outcome), so the SNR estimation is not necessary. For LDPC codes, the simplified decoding algorithm is known as uniformly most powerful BP-based (UMP-BP) algorithm [8] (or min-sum algorithm [6], [9]), and for TCs max-logAPP.

Sub-optimal calculation of the L-values unavoidably leads to performance degradation of the decoder due to two main reasons. First of all, a suboptimal calculation of the L-values produces information lost. Secondly, in the decoding process, sub-optimally calculated L-values are converted (implicitly or explicitly) into probabilities of bits as if they were optimal L-

Research supported by the Swedish Research Council, Sweden (under research grant #2006-5599), by the Government of Canada (under Graduate Students' Exchange Program (GSEP)), by NSERC, Canada (under research grant #249704-07), and by Fondecyt, Chile (project PBCT-ACT-11/2004).

values. While the former effect has no remedy (no processing of sub-optimal L-values can regenerate the information loss), it is possible to map the sub-optimal L-values into L-values that can be interpreted in terms of probabilities. Seeking for a compensation of the undesirable performance loss, it was found that linearly scaling the sub-optimal L-values improves the performance of the decoding algorithms. This linear scaling has evolved as a heuristic compensation, with almost no supporting theory. This idea was first proposed in [10], and next studied in various scenarios [11]–[17]. This correction can also be applied to optimally calculated L-values (optimum decoding algorithms), because after some iterations, the L-values are not independent due to the presence of cycles in the graph (cf. for example [13]–[15], [18]).

It is well understood that the optimum scaling factor depends on the SNR, the code (graph) structure, and the iteration number. Due to the obvious complexity of finding analytically the optimum scaling coefficients, numerical simulations are mostly used. Most of the existing approaches are based on an exhaustive search of the scaling factor in order to minimize the BER (e.g., [15], [18], [19]), which requires extensive simulations. Such an approach is valid but does not offer any explanation about why the correction via linear scaling produces improvement in the performance. The understanding of the mechanism underlying the correction principle is necessary not only to satisfy the fundamental curiosity, but also may provide new methods to correct the L-values.

Attempts to find the linear scaling factors without relying on the BER obtained from simulations also appeared in the literature. For example, [20] adjusted the mean of the sub-optimal L-values to match that of the optimal ones, while [10] adjusted the ratio of the mean and the variance to satisfy the so-called consistency condition assuming that the probability density function (pdf) of the L-values is Gaussian. A formal approach based on the analysis of the empirical pdf (histograms) of the L-values was presented in [19]. Nevertheless, the corrective factors obtained for TCs were found in [19] minimizing the BER via extensive simulations falling short of demonstrating the direct relationship between the proposed formalism and the improvement in the decoding process.

In this paper we propose to find a bijective mapping to convert suboptimal L-values into L-values that can be interpreted in terms of bits' probabilities. Using numerical examples, we demonstrate that the proposed mapping can be well approximated using a linear function which explains—using a theoretical basis—why the linear scaling factor widely used in the literature yields a very good performance. We show that for turbo codes the scaling factors found closely follow the optimum values which translates to a close-to-optimal BER performance. Moreover, for LDPC codes, the proposed method allows us to outperform the method known in the literature.

II. Data Model and Correction of L-values

We analyze here the decoding based on the iterative exchange of the L-values defined for the corresponding bits c. We will use two well-known examples of iterative decoding: turbo decoding of parallel concatenated convolutional codes (TCs), and message passing applied in the case of the LPDC codes.

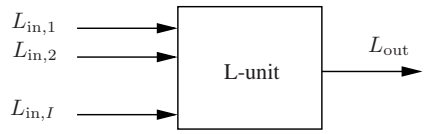

Figure 1. The L-values are calculated in the L-units using the input L-values obtained from the channel outcome or other L-units.

The decoding requires in the first instance calculation of the L-values from the the vector of observations (channel outcome) $\mathbf{r}$

$$L = \log \frac{\mathbb{P}(c=1|\mathbf{r})}{\mathbb{P}(c=0|\mathbf{r})} = \log \frac{\mathrm{p}(\mathbf{r}|c=1)}{\mathrm{p}(\mathbf{r}|c=0)}, \tag{1}$$

where the last term in (1) is obtained through Bayes' rule assuming $\mathbb{P}(c=b) = 1/2$ where $b \in \{0,1\}$, and that the conditional distribution $\mathrm{p}(\mathbf{r}|c=b)$ is known from the transmission model.

The L-values in (1) are sufficient statistics for $\mathbf{r}$, so the probability for the corresponding bits c may be calculated based on the L-values as

$$\mathbb{P}(c=b|\mathbf{r}) = \mathbb{P}(c=b|L) = \frac{\mathrm{e}^{b\cdot L}}{1+\mathrm{e}^{L}}. \tag{2}$$

Since the L-values are a function of the random channel outcome $\mathbf{r}$, they are also random variables characterized by their conditional pdf $\mathrm{p}_L(\lambda|c=b)$. In the following definition, we formalize the so-called *consistency condition* [21, Sec. III] which is the base for the methodology presented in this paper.

Definition 1 (Consistency Condition): A random variable $L \in \mathbb{R}$ is said to be *consistent* if

$$\log \frac{\mathrm{p}_L(\lambda|c=1)}{\mathrm{p}_L(\lambda|c=0)} = \lambda \quad \forall \lambda. \tag{3}$$

Consistent L-values based on the previous definition are also called *true* L-values, and can be converted into probabilities using (2).

The L-values of the coded bits obtained from the channel outcome $\mathbf{r}$ are used to obtain an approximation of the maximum-likelihood solution of the information bits. This is done propagating the L-values through the graph according to the sum-product algorithm. In the graph's nodes the new L-values are calculated by *L-units* (called also *box functions* in [19]) as shown in Fig. 1, where the input L-values $L_{\mathrm{in},i}$ are obtained from the channel outcome $\mathbf{r}$ or from other L-units. The L-unit correspond to the addition of bits in the Tanner graph defining the code and produces the extrinsic information for the corresponding bit c.

In the ideal case each L-unit produces outputs using

$$L_{\mathrm{out}} = \log \sum_{i=1}^{I} \mathrm{e}^{L_{\mathrm{in},i}} \tag{4}$$

which may be simplified to avoid the nonlinear functions involved in (4). In particular, applying the so-called max-log approximation, we obtain

$$L_{\text{out}} = \max_i L_{\text{in},i}. \tag{5}$$

If (5) is used in an L-unit, its outcome will not be consistent; the resulting L-values are corrupted and cannot be directly transformed into probabilities using (2). In the following theorem we formally define a mapping between a corrupted L-value and a new L-value that fulfills the consistency condition.

Theorem 1: A random variable $L' = f(L)$ obtained from a random variable L, with $L, L' \in \mathbb{R}$, using a bijective mapping $f : \mathbb{R} \to \mathbb{R}$ defined by

$$f(\lambda) = \log \frac{\mathrm{p}_L(\lambda|c=1)}{\mathrm{p}_L(\lambda|c=0)} \quad \forall \lambda \in \mathbb{R} \tag{6}$$

fulfills the consistency condition (3).

Proof: Since $f(\lambda)$ is bijective, its inverse $f^{-1}(\lambda)$ exists so the pdf of the variable L' is given by [22, Ch. 5.2]

$$\mathrm{p}_{L'}(\lambda|c=b) = \mathrm{p}_L(f^{-1}(\lambda)|c=b) \cdot \frac{d}{d\lambda} f^{-1}(\lambda). \tag{7}$$

Using (7) in (6) yields the consistency definition in (3). ■

Using this theorem, the knowledge of the pdf of the non-consistent L-values is sufficient to obtain a new true L-value using $f(\lambda)$. The only limitation in applying the mapping $f(\lambda)$ is the knowledge of the pdf of the L-values, which is in general not known, but it may be acquired through simulations (via histograms). For a given SNR, we proceed, therefore, in the following steps

1) We acquire the histograms of the L-values sent to other L-units in order to estimate $\mathrm{p}_L(\lambda|c)$.
2) We obtain $f(\lambda)$ using (6) and the previously estimated pdf.
3) We modify the decoder to take into account the correction of L using $f(\lambda)$.

For the LDPC codes, the previous steps can be applied to the L-values computation at the check nodes (at the variable nodes the L-values are simply added). For the TC, the procedure can be applied to the extrinsic L-values associated to the systematic bits, which are exchanged between both constituent APP decoders.

In Fig. 2 we show examples of the resulting estimated pdf obtained for an LDPC code after the first decoding iteration. The code used to generate this figure is a regular LDPC code defined in Sec. III-A for two check-nodes degree $\rho = 6$ and $\rho = 12$ and the UMP-BP decoder. The histograms for the extrinsic information after the first half iteration of a TC using max-logMAP for two code rates is also shown.

Using the densities of Fig. 2, which are symmetric, i.e., $\mathrm{p}_L(\lambda|c=1) = \mathrm{p}_L(-\lambda|c=0)$, $f(\lambda)$ in (6) can be calculated. This function is presented in Fig. 3 (markers) for the four different cases. The next step is to approximate $f(\lambda)$ using some function which is simple to implement. From the shapes of $f(\lambda)$ in Fig. 3, a linear approximation clearly seems to be a good choice., i.e., $f(\lambda) \approx \alpha \cdot \lambda$. Any other function providing a good fitting can be used as an approximation of $f(\lambda)$, for example, a low-order polynomial. The linear approximation however, has the undeniable advantage of being simple to implement, and yet provides a very good approximation of $f(\lambda)$. With the linear approximation $f(\lambda) \approx \alpha \cdot \lambda$, we are in fact showing that α is the well-known (heuristic) linear scaling factor used in the literature. Moreover, if the methodology

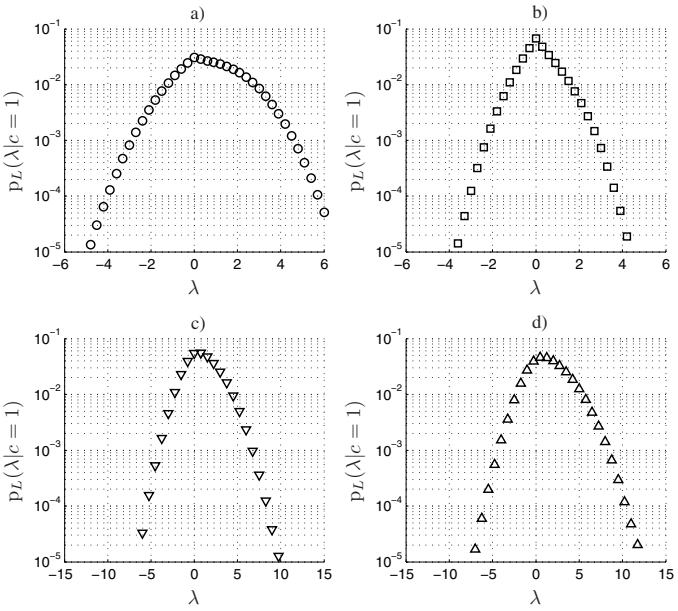

Figure 2. Histograms of the L-values after the first decoding iteration of the LDPC code using UMP-BP for $E_\mathrm{b}/N_0 = 2.5$ dB at the check nodes for a) $\rho = 6$, and b) $\rho = 12$. Histograms of the extrinsic L-values for the first half iteration of the TC for $E_\mathrm{b}/N_0 = 1.0$ dB using max-logMAP for c) $R = 1/2$, and d) $R = 1/3$.

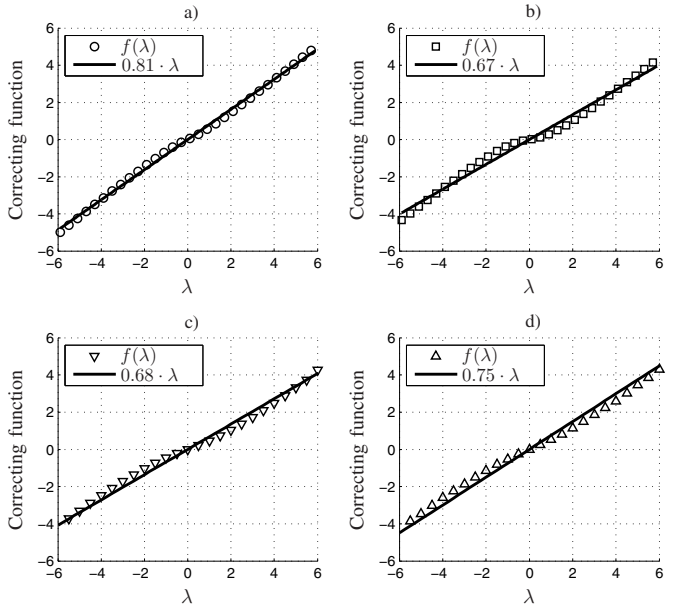

Figure 3. The correcting function $f(\lambda)$ defined in (6) obtained from the histograms shown in Fig. 2 (markers) and its linear approximation (solid line).

presented above is used, the calculation of this factor does not require an exhaustive search.

For both cases (TC and LDPC) we may repeat the above procedure for any L-value in the decoder, thus making α dependent on the number of iterations. However, this becomes more and more complicate through the iterations since the tails of the densities become more and more difficult to estimate. Moreover, our simulation showed that the BER results obtained using an iteration-dependent α are very similar to the ones obtained using the same α calculated in the first iteration. Consequently we use this approach for the numerical examples presented in the next section.

III. Numerical Examples

In this section we compare the decoding based on the exhaustive search of the scaling factors α with the results based on the methodology outlined in the previous section. for the simulations, the bits are transmitted over an AWGN channel using binary phase-shift keying. For the LDPC code the decoder performs 1000 iterations, and for the TC the decoder performs 10 iterations.

A. LDPC codes

We use the two regular rate $R = 1/2$ LDPC Gallager codes $(504, 252)$ and $(1008, 504)$ with check-nodes degree $\rho = 6$ and variable-nodes degree $\nu = 3$. In fact we note that that value of ν does not affect the pdf of the L-values at the check nodes in the first iteration. The parity check matrices were obtained from [23]. The L-values at the check nodes are calculated using four different approaches: a) BP, i.e., (4) is used, b) UMP-BP, i.e., (5) is used, c) UMP-BP-CF, i.e., UMP-BP is used with a scaling factor α obtained as shown by Chen and Fossorier [8], and d) UMP-BP-CO, i.e., UMP-BP is used with scaling factor found using the proposed methodology, that is scaling the L-value to make it satisfy the consistency condition.

In Fig. 4 we present the BER as a function of the scaling factor α for two lengths of the LDPC code and four different SNRs. The optimum factors $\hat{\alpha}$ are found at the minima of each curve where we note that the minimum is not well defined due to the randomness of the Monte-Carlo results and the flattening of the curves close to their minimum (e.g., for $\alpha = \hat{\alpha} \pm 0.05$, the resulting BER is practically the same). We note immediately that appropriate scaling can significantly lower the BER when comparing to the case $\alpha = 1$ (UMP-BP).

To obtain the curves, extensive numerical simulations were performed, where the complexity increases if the targeted BER is reduced. Finally, note that the optimum factors $\hat{\alpha}$ seem to be independent of the code-length. They depend, however, on the check-nodes degree as shown already in [8]. The optimal factors $\hat{\alpha}$ are contrasted in Fig. 5 with those obtained from the analysis of pdf. The approach we presented gives results very close to the optimum (the actual difference in the BER performance is negligible due to flattening of the curve). Moreover, using the consistency criteria the extensive numerical simulations are avoided.

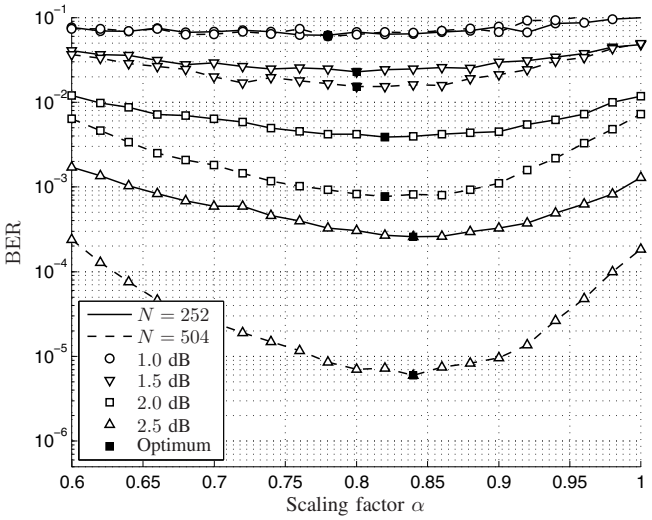

Figure 4. Exhaustive search of the optimum scaling factor α for LDPC codes.

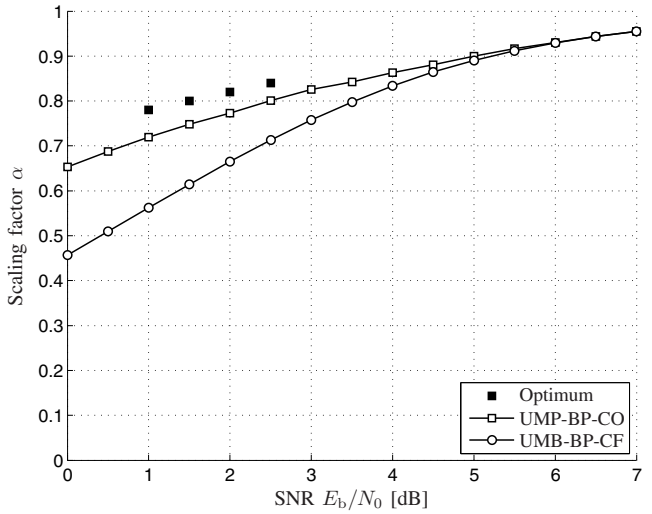

Figure 5. Scaling factors for LDPC codes based on the exhaustive grid search (optimum), on the consistency condition, and on the approach of [8].

On the other hand the results obtained using the approach of Chen and Fossorier [8] are quite far from the optimum which results in a visible performance degradation which is well illustrated in Fig. 6 and Fig. 7. We note also that the algorithm BP produces results slightly worse than UMP-BP for very low BER. This is not entirely surprising as for a large number of iterations, and due to the presence of cycles in the graph, the L-values are not independent and algorithms which are theoretically suboptimal may outperform those based on the "optimal" calculation of L-values; this effect has been observed already in [18], [24].

B. Turbo Codes

We use a TC formed by the parallel concatenation of two identical recursive systematic convolutional encoders, each of

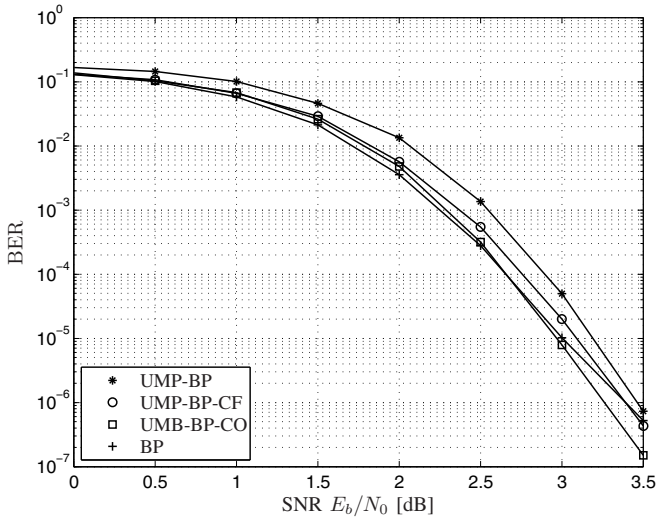

Figure 6. BER for the $(504, 252)$ LDPC code and different algorithms.

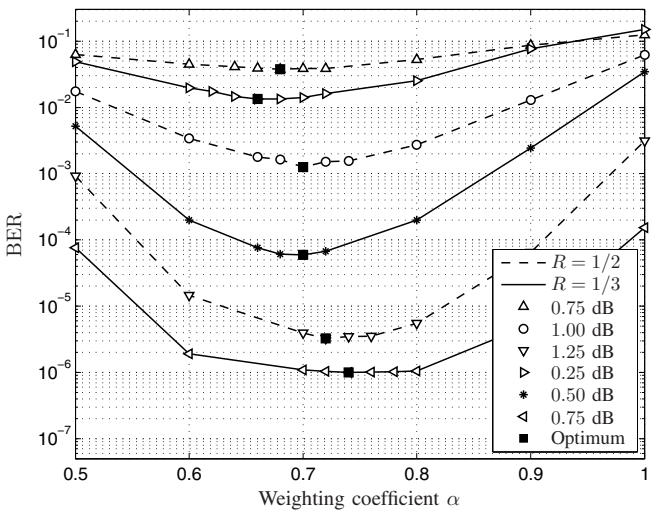

Figure 8. Exhaustive search for the optimum scaling factors α for TCs.

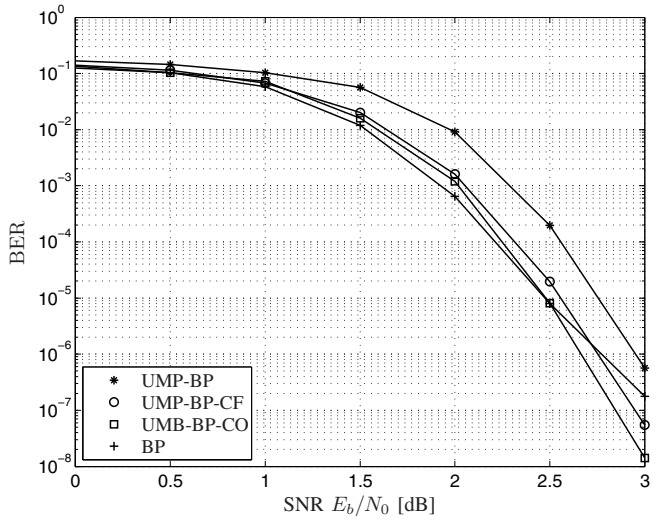

Figure 7. BER for the $(1008, 504)$ LDPC code and different algorithms.

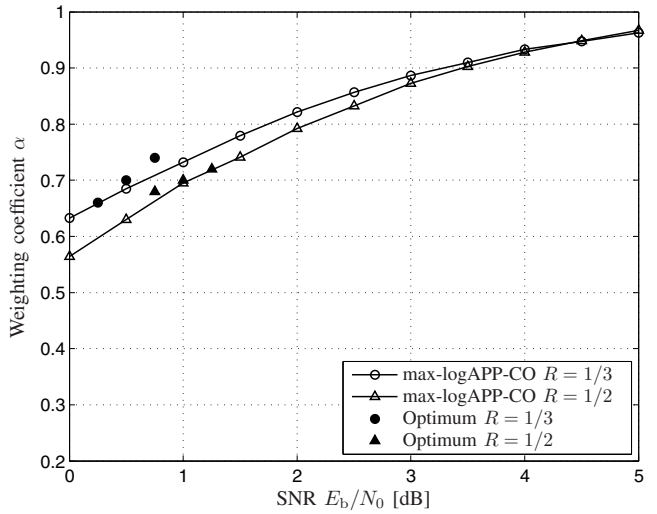

Figure 9. Correction factors for turbo codes based on the exhaustive grid search (optimum) and based on the consistency condition for different code rates.

them defined by their generator polynomials $(1, 13/15)_8$. Two block lengths $N = 512$ and $N = 5120$ are considered, as well as two code rates $R = 1/3$ (no puncturing) and $R = 1/2$ (alternated puncturing of the parity bits). Four different approaches are considered: a) logAPP, i.e., the calculation is done optimally based on (4), b) max-logAPP, i.e., all operation of type (4) within the original APP are replaced with (5), c) max-logAPP-0.7, i.e, max-logAPP scaling the resulting L-values with $\alpha = 0.7$ as recommended by [13], and d) max-logAPP-CO, i.e, max-logAPP with a scaling factor α obtained through the proposed methodology.

In Fig. 8 we present the BER as a function of the scaling factor α for the two TCs analyzed. The same procedure was applied to both block lengths, however, the optimum values of α obtained were identical (for the same SNR and R). We conjecture then that the consistency property of the L-values, and therefore the optimum scaling factor, does not depend on the block length but rather on the code's structure, the puncturing pattern (code rate), and the SNR.

In Fig. 9 we present the scaling factors for the TC described above comparing them with those obtained through our approach.

Using the results of Fig. 8 where the minimum of the BER curve is very flat in the vicinity of the minimum, we should expect only a small performance degradation when using the coefficients calculated from the consistency criterion. This is indeed confirmed by results shown in Fig. 10 and Fig. 11. Using max-logAPP-CO the results are practically the same as using constant factor $\alpha = 0.7$ in max-logAPP-0.7. The max-logAPP algorithms with scaling factor are able to outperform the APP algorithm as it has also been observed in [18].

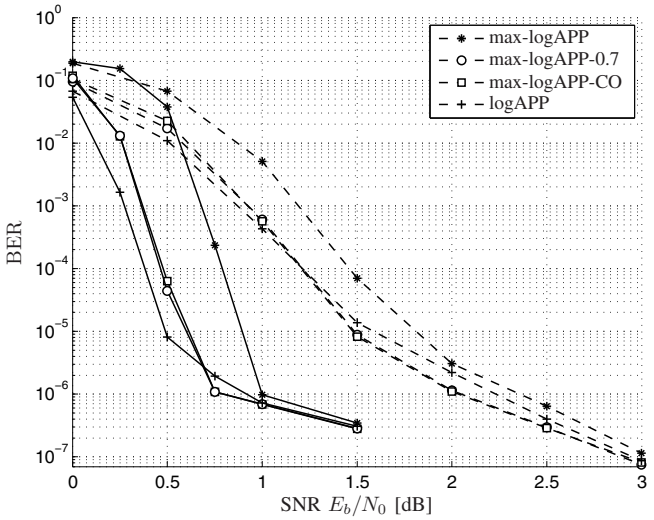

Figure 10. Simulation results for $R = 1/3$ TCs: BER for and different decoding algorithms for $N = 512$ (dashed lines) and for $N = 5120$ (solid lines).

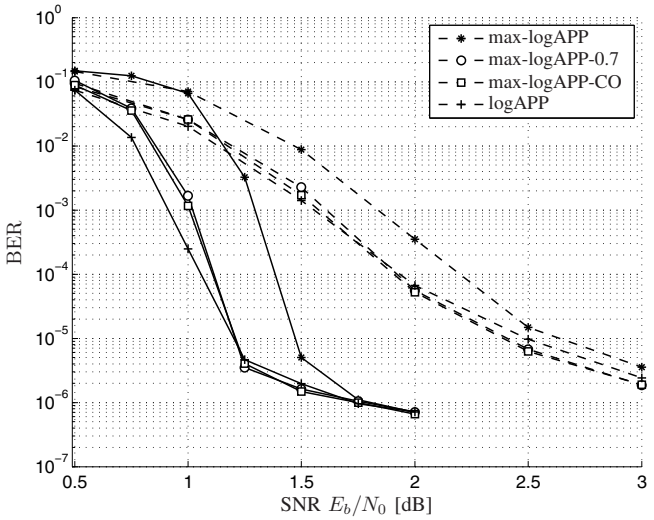

Figure 11. Simulation results for $R = 1/2$ TCs: BER for and different decoding algorithms for $N = 512$ (dashed lines) and for $N = 5120$ (solid lines).

IV. Conclusions

In this paper we presented a method to improve the performance of suboptimal iterative decoding algorithms. The method is based on converting corrupted L-values to L-values that fulfill the consistency condition, and consequently, can be correctly interpreted as bits' probabilities. We applied the proposed methodology to turbo and LDPC codes, and we showed that the scaling factors found closely follow the optimum values, which translates into a close-to-optimal BER performance. The proposed method avoids extensive simulations and provides a formal explanation of why the well-known (heuristic) linear scaling factor of the sub-optimal L-values yields improvement of the decoders' performance.

References

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in *International Conference on Communications, ICC 1993*, Geneva, Switzerland, May 1993, pp. 1064–1070.

[2] R. Gallager, "Low-density parity-check codes," *IRE Trans. Information Theory*, pp. 21–28, Jan. 1962.

[3] D. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 18, pp. 399–431, Mar. 1999.

[4] H. A. Loeliger, "An introduction to factor graphs," *IEEE Signal Process. Mag.*, vol. 21, no. 1, pp. 28–41, Jan. 2004.

[5] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA: Morgan Kaufnann, 1988.

[6] W. E. Ryan, *"An Introduction to LDPC Codes", in CRC Handbook for Coding and Signal Processing for Recording Systems*, B. Vasic, Ed. CRC Press, 2004.

[7] L. J. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimum decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, pp. 284–287, Mar. 1974.

[8] J. Chen and M. Fossorier, "Near optimum universasl belief propagation based decoding of low-density parity check codes," *IEEE Trans. Commun.*, vol. 50, no. 3, pp. 406–414, 2002.

[9] N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation, Linköping University, Linköping, Sweden, 1996.

[10] L. Papke, P. Robertson, and E. Villebrun, "Improved decoding with the SOVA in a parallel concatenated (turbo-code) scheme," in *International Conference on Communications, ICC 96*, vol. 1, Jun 1996, pp. 102–106.

[11] R. M. Pyndiah, "Near-optimum decoding of product codes: Block turbo codes," *IEEE Transactions on Communications*, vol. 46, no. 8, pp. 1003–1010, Aug 1998.

[12] S. Crozier, K. Gracie, and A. Hunt, "Efficient turbo decoding techniques," in *11th International Conference on Wireless Communications (Wireless 99)*, Calgary, Alberta, Canada, Jul. 1999, pp. 187–195.

[13] J. Vogt and A. Finger, "Improving the max-log-map turbo decoder," *IEEE Electronic Letters*, vol. 36, no. 23, pp. 1937–1939, Nov. 2000.

[14] C. Chaikalis, J. Noras, and F. Riera-Palou, "Improving the reconfigurable SOVA/log-MAP turbo decoder for 3GPP," in *IEEE / IEE International Symposium on Communication Systems, Networks and DSP*, Stafford, UK, Jul. 2002.

[15] Y. Ould-Cheikh-Mouhamedou, P. Guinand, and P. Kabal, "Enhanced Max-Log-APP and enhanced Log-APP decoding for DVB-RCS," in *International Symposium on Turbo Codes*, May 2003, pp. 136–145.

[16] K. Gracie, S. Crozier, and P. Guinand, "Performance of an mlse-based early stopping technique for turbo codes," in *IEEE Vehicular Technology Conference 2004, VTC-2004 Fall*, vol. 3, 2004, pp. 2287– 2291.

[17] K. Gracie, A. Hunt, and S. Crozier, "Performance of turbo codes using mlse-based early stopping and path ambiguity checking for input quatized to 4 bits," in *4*th *International Symposium on Turbo Codes & Related Topics*, Munich, Germany, April 2006.

[18] C. X. Huang and A. Ghrayeb, "A simple remedy for the exaggerated extrinsic information produced by the sova algorithm," *IEEE Journal on Wireless Communications*, vol. 5, no. 5, pp. 996–1002, May 2006.

[19] M. van Dijk, A. Janssen, and A. Koppelaar, "Correcting systematic mismatches in computed log-likelihood ratios," *Eur. Trans. on Telecommun.*, vol. 14, no. 3, pp. 227–224, 2003.

[20] H. Chen and A. M. Haimovich, "Low complexity turbo mimo systems," in *2002 Conf. on Information Science and Systems*, Princeton University, Mar. 2002.

[21] M. Tüchler, "Design of serially concatenated systems depending on the block length," *IEEE Trans. Commun.*, vol. 52, no. 2, pp. 209–218, Feb. 2004.

[22] A. Papoulis and U. Pillai, *Probability, Random Variables and Stochastic Processes*, 4th ed. McGraw-Hill, 2002.

[23] D. MacKay, "Encyclopedia of sparse graph codes," available online at http://www.inference.phy.cam.ac.uk/mackay.

[24] M. Yazdani, S. Hemati, and A. Banihashemi, "Improving belief propagation on graphs with cycles," *IEEE Commun. Lett.*, vol. 8, no. 1, pp. 57–59, Jan. 2004.