Lossy Source Coding using Belief Propagation and Soft-Decimation Over LDGM Codes

Daniel Castanheira
Instituto de Telecomunicações
Aveiro University
Campo Universitário, Aveiro, 3810-193, Portugal
Email: dcastanheira@av.it.pt

Atilio Gameiro
Instituto de Telecomunicações
Aveiro University
Campo Universitário, Aveiro, 3810-193, Portugal
Email: amg@ua.pt

***Abstract*—This paper focus on the lossy compression of a binary symmetric source. We propose a new algorithm for binary quantization over low density generator matrix (LDGM) codes. The proposed algorithm is a modified version of the belief propagation (BP) algorithm used in the channel coding framework and has linear complexity in the code block length. We also provide a common framework under which the proposed algorithm and some previously proposed algorithms fit. Simulation results show that our scheme achieves close to state-of-the-art performance with reduced complexity.**

I. Introduction

Lossy source coding is important in many applications, namely dirty paper coding [1], information hiding [2], data embedding [3] and watermarking. However, even if for the channel coding problem there exist efficient algorithms to explore the sparse graph structure of sparse graph codes, that achieve astonishing performance, namely the ones belonging to the family of message passing algorithms [4], [5], the same is not true for the lossy source coding problem. In [6] the authors have shown that LDGM codes, as duals of low density parity check (LDPC) codes, can saturate the rate-distortion bound of the binary erasure quantization (BEQ) problem, the dual of the binary erasure symmetric channel (BEC) problem, with the help of a modified BP algorithm. Since this pioneering work, LDGM codes have been extensively used for lossy source coding. Namely in [7] the authors propose a survey propagation (SP) based iterative quantization algorithm and in [8] the authors propose a BP based iterative quantization algorithm, to compress a binary symmetric source. However both use a decimation process to help the respective algorithm to converge. Thus the overall computational complexity is $\mathcal{O}(N^2)$, [9], where N is the codeword length. To overcome the complexity bottleneck, in [9], the authors propose a linear complexity algorithm to do lossy source coding with the help of LDPC codes over GF(q). However the use of higher order fields substantially increase the computational complexity, in comparison to the binary field. In contrast we propose to do lossy source coding with a binary LDGM code and with a modified BP algorithm, with computational complexity $\mathcal{O}(N)$, since no decimation is used.

This paper is organized as follows: section II provides some background about LDGM codes and about the lossy source coding problem, to the reader. In section III we describe and derive the proposed algorithm for lossy compression. Next in section IV the performance of the proposed scheme is evaluated, by numerical simulations, and finally we conclude the article in section V.

II. Framework

A. Problem Description

In this paper we focus on the lossy compression of a binary symmetric source. For a given realization $\mathbf{s} \in \{0,1\}^N$ of a Ber$(1/2)$ source $\mathbf{S}$ we want to map it to an index $\mathbf{x} \in \{0,1\}^K$ such that the approximate reconstruction of $\mathbf{s}$ is possible within a given fidelity criterion. $R = K/N$ is the compression rate. The fidelity measure used here and the most commonly used in this area, for a binary source, is the Hamming distance between the source sequence $\mathbf{s}$ and the reconstructed sequence $\hat{\mathbf{s}}$:

$$d(\mathbf{s}, \hat{\mathbf{s}}) = \frac{1}{n} \sum_{i=1}^{n} |s_i - \hat{s}_i| \tag{1}$$

The objective of lossy compression is to minimize the average distortion $D = \mathbb{E}[d(\mathbf{S}, \hat{\mathbf{S}})]$. For such a distortion measure the asymptotic limit, know as rate-distortion function, is well known [10] and is given by $R(D) = 1 - h(D)$, for $D \in [0, 1/2]$ and zero otherwise, where $h(.)$ denotes the binary entropy function.

B. LDGM Code Description and Notation

LDGM codes are the most commonly used type of codes for the lossy source coding problem. They are duals of LDPC codes. Their performance in the channel coding problem is limited. Due to their poor distance properties, they exhibit an error floor, that is independent of the considered block-length. Even though, this problem can be easily controlled with proper concatenation of two codes [11]. On the other hand, for the lossy source coding problem, their performance is very good, as can be attested by the empirical results obtained in [7] and [8].

A LDGM code can be represented both by its generator matrix $\mathbf{G} \in \{0,1\}^{N \times K}$ and by the associated factor graph $\mathcal{G} = (V, C, E)$. Where the sets $V = \{1, \ldots, K\}$, $C = \{1, \ldots, N\}$ and $E = \{\ldots, (a, i), \ldots\}$ denote the information bit nodes, the check nodes and the edges connecting them,

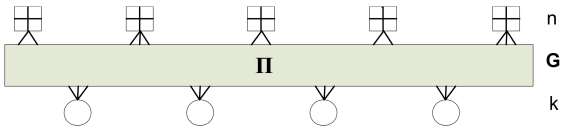

Fig. 1. LDGM code factor graph representation.

respectively. We use the variables $a, b, c \in C$ to denote check nodes and variables $i, j, k \in V$ to denote information bits. We define the sets $C(i) = \{a \in C | (a, i) \in E\}$, $V(a) = \{i \in V | (a, i) \in E\}$ and use the symbol $\setminus$ to denote the set subtraction operator. A check node $a \in C$ connects to an information bit $i \in V$, $(a, i) \in E$, if $\mathbf{G}_{a,i} = 1$. In Fig. 1 we depict, as an example, the factor graph of a given LDGM code with 5 check nodes, 4 information bits and 5 source bits, randomly connected between them. In this figure, the white circles denote the information bits, the squares represent the check nodes and by $\mathbf{\Pi}$ we mean a uniform drawn permutation.

The connection structure between check nodes and information bits, in a LDGM code, is specified by the correspondent check and information bit node degree distributions, from the edge perspective, (ρ, λ), $\rho(x) = \sum_i \rho_i x^{i-1}$ and $\lambda(x) = \sum_i \lambda_i x^{i-1}$. Where ρ_i and λ_i denote the portion of all edges connected to check nodes and information bits with degree i, respectively. The degree of a check node is equal to the number of connected information bits (associated source bit not taken into account), which is the same as the corresponding number of entries equal to 1 in row i of $\mathbf{G}$. For a LDGM code, $\mathcal{C}$, defined by the generator matrix $\mathbf{G}$, and for an index $\mathbf{x}$ the corresponding reconstructed source sequence is given by $\hat{\mathbf{s}} = \mathbf{G}\mathbf{x}$. Which due to the sparse structure of the corresponding LDGM code can be computed in $\mathcal{O}(N)$ time, for a given index $\mathbf{x}$.

III. Derivation of the Proposed Algorithm

Almost all previously proposed algorithms for doing lossy source coding rely on a decimation step to help the iterative message passing algorithm, like BP, to converge. Namely, this type of algorithms can be divided into two phases, one where BP runs for a given number of iterations and another where the most biased information bits are decimated. Even if the resultant performance of such algorithms is very good, in practice, their inherent computational complexity is $\mathcal{O}(N^2)$. One exception to this rule is the algorithm proposed in [9], [12], the reinforced BP (RBP). This algorithm does a kind of soft decimation by reinforcing the beliefs of each information bit at each iteration. Our proposed algorithm is based on the same principle, but it is applied to LDGM codes instead of LDPC codes and we also show a connection between "hard" and "soft" decimation.

In the following paragraphs we explain how to transform the decimation step, of the previously proposed algorithms, in a soft decimation step, that is performed within each iteration. We also show how to use this simple transformation to recover the RBP algorithm as a soft version of the Bias Propagation Algorithm (BiP), [8]. This transformation results in just a slightly modification to the BP updating rules.

In the LDGM code approach to lossy source coding, the encoding phase amounts to mapping a source sequence $\mathbf{s} \in \{0,1\}^N$ to an index $\mathbf{x} \in \{0,1\}^K$, such that the Hamming distortion $d(\mathbf{G}\mathbf{x}, \mathbf{s})$ is minimized. Let us define the following equivalent conditional probability distribution:

$$P(\mathbf{x}|\mathbf{s}) = \frac{1}{Z} e^{-\gamma d(\mathbf{G}\mathbf{x}, \mathbf{s})} = \frac{1}{Z} \prod_{a \in C} e^{-\gamma d(\mathbf{G}_a \mathbf{x}, \mathbf{s}_a)} \tag{2}$$

where Z is a normalizing constant, $\mathbf{G}_a$ denote row a of $\mathbf{G}$ and γ is a parameter that is tuned via simulation to get as good performance as possible. It is not difficult to see, from equation (2), that the most probable codeword is also the one that minimizes the corresponding Hamming distortion. Indeed, the best assignment for bit x_i is obtained from the corresponding bit marginal value:

$$\begin{aligned} x_i &= arg \max_{x_i \in \{0,1\}} P(x_i|\mathbf{s}) = arg \max_{x_i \in \{0,1\}} \sum_{\sim x_i} P(\mathbf{x}|\mathbf{s}) \\ &= arg \max_{x_i \in \{0,1\}} \sum_{\sim x_i} \prod_{a \in C} \Psi(\mathbf{x}_{V(a)}, \mathbf{s}_a) \end{aligned} \tag{3}$$

The last line can be obtained by taking into account the special structure of $P(\mathbf{x}|\mathbf{s})$, of the LDGM code. $P(\mathbf{x}|\mathbf{s})$ can be factorized in a product distribution, where each element is equal to $\Psi(\mathbf{x}_V(a), \mathbf{s}_\mathbf{a})$ and represent the local constraint of check node a. The marginal value of bit i, given by equation (3), can be efficiently computed, if the BP/Sum-Product algorithm is used. However, unlike in the channel coding problem, where the received codeword is normally at a short distance from a codeword, in the lossy source coding problem the interference sequence is likely to be equidistant from more than one codeword, producing belief values, about the bit marginals, close to $1/2$. Consequently, the locally operating algorithm can get confused about the direction to proceed. The usual procedure to overcome the aforementioned problem is to decimated the most biased bits, after running the BP algorithm, and to repeat the previous two steps (BP and decimation) until all bits get decimated.

A. Belief Propagation

After some algebraic manipulations, it is not difficult to express the BP update equations, for the lossy source coding problem, as follows, [8]:

From variable to check:

$$R_i^{n+1} = \prod_{b \in C(i)} R_{bi}^n, \qquad R_{ia}^{n+1} = \prod_{b \in C(i) \setminus a} R_{bi}^n \tag{4}$$

From check to variable:

$$R_{ai}^{n+1} = \frac{1 - S_{ai}^n}{1 + S_{ai}^n}, \quad S_{ai}^n = (-1)^{s_a} \frac{1 - e^{-2\gamma}}{1 + e^{-2\gamma}} \prod_{j \in V(a) \setminus i} B_{ja}^n \tag{5}$$

Variable Bias:

$$B_i^n = \frac{1 - R_i^n}{1 + R_i^n}, \qquad B_{ia}^n = \frac{1 - R_{ia}^n}{1 + R_{ia}^n} \tag{6}$$

where R_{ia}^n, R_{ai}^n and B_{ia}^n denote the message sent from variable node i to check node a, the message sent from check node a to variable node i and the variable i bias, without taking into account the information coming from check node a, at iteration n, respectively. By R_i^n and B_i^n we represent the ratio between the probability of variable i being equal to one and the probability of being equal to zero, and the variable i bias, respectively. The previous BP update equation can be easily converted to the log-domain, using the hyperbolic tangent function definition:

From variable to check:

$$\bar{R}_i^{n+1} = \sum_{b \in C(i)} \bar{R}_{bi}^n, \qquad \bar{R}_{ia}^{n+1} = \sum_{b \in C(i) \setminus a} \bar{R}_{bi}^n \tag{7}$$

From check to variable:

$$\bar{R}_{ai}^{n+1} = 2(-1)^{s_a+1} tanh^{-1} \left[\beta \prod_{j \in V(a) \setminus i} B_{ja}^n \right] \tag{8}$$

Variable Bias:

$$B_i^n = -tanh\left(\frac{\bar{R}_i^n}{2}\right), \qquad B_{ia}^n = -tanh\left(\frac{\bar{R}_{ia}^n}{2}\right) \tag{9}$$

where the top bar denotes the transformation $\bar{x} = log(x)$ and $\beta = tanh(\gamma)$.

B. Hard-Decimation

The decimation step implies that the corresponding code factor graph can be reduced [8], or equivalently that the decimated information bits always send the message $+\infty$ or $-\infty$, to the neighboring check nodes. This fact can be easily incorporated into the corresponding decimated variable node update equation or into their neighboring check nodes update equations. To do that we just need to add to each check node or to each variable node or to both update equations an indicator function, $I_d(\alpha_i, t)$, taking the value 0 (bit still not decimated), $-\infty$ (bit decimated to 0) or $+\infty$ (bit decimated to 1):

$$I_d(\alpha_i, t) = \begin{cases} -\infty, & \text{if } \alpha_i \leq -t \\ 0, & \text{if } -t \leq \alpha_i < t \\ +\infty, & \text{if } t \leq \alpha_i \end{cases} \tag{10}$$

where α_i can be any parameter, normally a bit bias, and t denotes a threshold parameter, after which the bit gets decimated. In the BiP algorithm $\alpha_i = -B_i$. It should be emphasized here that the decimation step is not as simple as that. For example, the decimation, in the BiP algorithm, only decimates a fixed number of information bits, at each round. More precisely, it decimates the most biased m bits, even if there are more bits that respect the threshold t. However, as we will see in the next sections, this approximation is sufficient to obtain very good results.

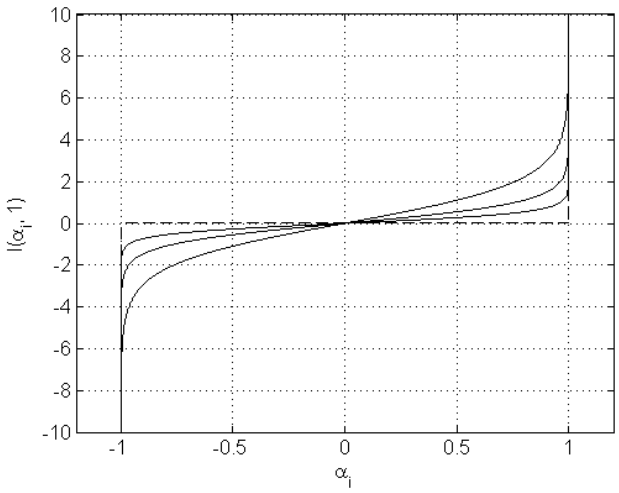

Fig. 2. The dashed line show the function $I_d(\alpha_i, 1)$ and the solid curves show $\hat{I}_d(\alpha_i, 1) = (2/\mu)tanh^{-1}(\alpha_i)$, for $\mu = \{1, 2, 4\}$. The curve for $\mu = 4$, gives the best approximation.

C. Soft-Decimation

The basic idea of our method is to approximate the indicator function I_d by the soft indicator function:

$$\hat{I}_d(\alpha_i, t) = \frac{2}{\mu}\, tanh^{-1}\left(\frac{\alpha_i}{t}\right),\ \alpha_i \in]-t, t[\tag{11}$$

where $\mu > 0$ is a parameter that sets the accuracy of the approximation. Figure 2 shows the function I_d and the approximation for several values of μ. As μ increases the approximation becomes more accurate. Assume that $\alpha_i = -B_i^n$ and that $t = 1$, since, for $t < 1$, the argument of the $tanh^{-1}(.)$ function can get higher than one ($-1 \leq B_i^n \leq 1$), and for that range of values the inverse tangent hyperbolic function is undefined. Consequently $\hat{I}_d(-B_i^n, 1) = (1/\mu)\bar{R}_i^n$. The same happens if we consider $\alpha_i = -B_{ia}^n$ and $t = 1$. More precisely the indicator function, that represents the decimation step, is replaced by a simple linear function. Therefore, if the indicator function is added at the variable node update equations and $\alpha_i = -B_i^n$, the BP update equations, with decimation included, can be expressed by:

From variable to check:

$$\bar{R}_i^{n+1} = \sum_{b \in C(i)} \bar{R}_{bi}^n, \qquad \bar{R}_{ia}^{n+1} = \frac{1}{\mu}\bar{R}_i^n + \sum_{b \in C(i) \setminus a} \bar{R}_{bi}^n \tag{12}$$

From check to variable:

$$\bar{R}_{ai}^{n+1} = 2(-1)^{s_a+1} tanh^{-1} \left[\beta \prod_{j \in V(a) \setminus i} B_{ja}^n \right] \tag{13}$$

Variable Bias:

$$B_i^n = -tanh\left(\frac{\bar{R}_i^n}{2}\right), \qquad B_{ia}^n = -tanh\left(\frac{\bar{R}_{ia}^n}{2}\right) \tag{14}$$

In this new formulation, we use a linear or 'soft' constraint function in place of a hard constraint, $I_d(.)$, like in the

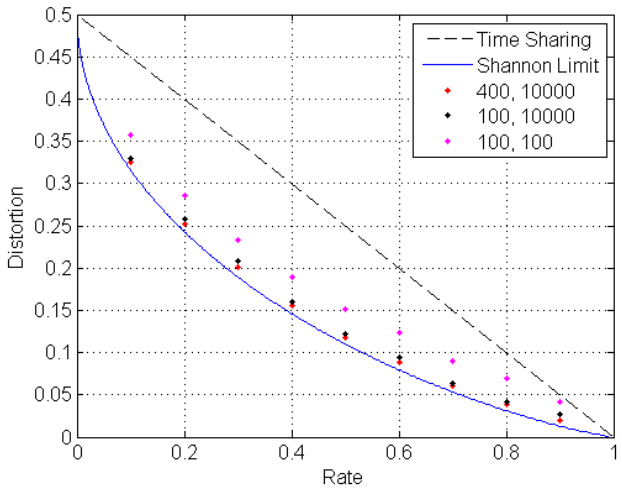

Fig. 3. Lossy source coding performance of the proposed algorithm for different source sequence lengths (100, 10000) and for different number of iterations, (100, 400).

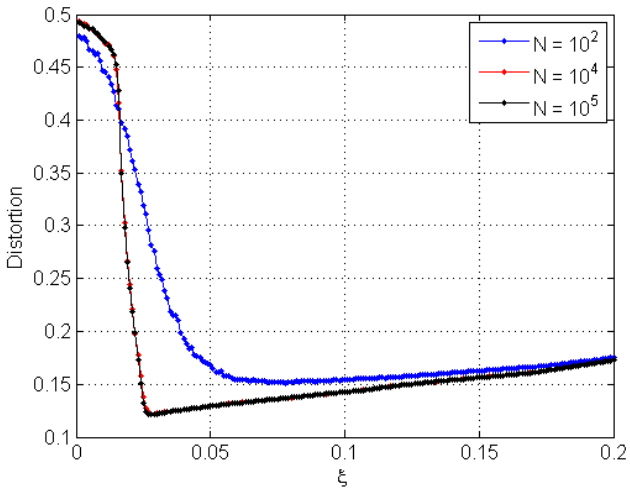

Fig. 4. Lossy source coding performance, for 100 iterations, as a function of ξ for a randomly generated graph with rate equal to 0.5 and for three codeword lengths $N = \{10^2, 10^4, 10^5\}$.

interior-point method [13]. A more careful look to this update equations show that, if $tanh(\gamma) = 1$, this are indeed the log-domain RBP update equations, used in [9], for doing lossy source coding, with LDPC codes over GF(q). On the other hand, if $\alpha_i = -B_{ia}^n$ and the indicator function is added to the check node update equations, the BP update equations, with decimation included, can be expressed by:

From variable to check:

$$\bar{R}_i^{n+1} = \sum_{b \in C(i)} \bar{R}_{bi}^n, \qquad \bar{R}_{ia}^{n+1} = \sum_{b \in C(i) \setminus a} \bar{R}_{bi}^n \tag{15}$$

From check to variable:

$$\bar{R}_{ai}^{n+1} = \frac{1}{\mu} \bar{R}_{ia}^n + 2(-1)^{s_a+1} tanh^{-1} \left[\beta \prod_{j \in V(a) \setminus i} B_{ja}^n \right] \tag{16}$$

Variable Bias:

$$B_i^n = -tanh\left(\frac{\bar{R}_i^n}{2}\right), \qquad B_{ia}^n = -tanh\left(\frac{\bar{R}_{ia}^n}{2}\right) \tag{17}$$

The main advantage of this new formulation is that the algorithm constrains the belief of the information bits, at each iteration, in the direction of the current bit belief, instead of constraining only the beliefs at the decimation step, allowing a refinement of the information bit beliefs, at each iteration. As a consequence all bits are decimated at once instead of being decimated, a fixed number of information bits, at each decimation step, as in [7] and [8]. Since, in the new algorithm, the decimation step is incorporated in the BP update equations and this amounts to just one more addition and multiplication, per edge, the overall computational complexity of the proposed algorithm is $\mathcal{O}(N)$, instead of $\mathcal{O}(N^2)$, as will be confirmed in section IV.

IV. Results

To access the performance of our algorithm we have simulated it for various code rates, source sequence lengths and different number of iterations. The modified BP update equations used in the simulations were equations (15), (16) and (17). The codes chosen were from degree distributions optimized for the BEC or binary symmetric channel (BSC), in light of LDPC channel coding and LDGM source coding duality, in the erasure case. All codes used in the simulations were generated randomly according to the correspondent degree distribution. No 4-cycles or higher order cycles have been removed from the corresponding factor graph, only double edges. In all results presented in this section we have considered that $\beta = (1 - \xi)/(1 + \xi)$ and that $\xi = 1/\mu$, even if we have the freedom to tune both β and μ parameters independently. This parametrization has been chosen due too the inherent good performance and since it simplifies the algorithm and the task of accessing its performance.

Concerning the information bits prior value, since no apriori information is available it should be initialized to zero. However since we are talking about a LDGM code and as is well known from the channel coding framework: if no degree 1 check nodes are available the BP algorithm never starts, it gets stuck. In [7] and [8] where a decimation step is available, this can be easily overcome since there we always have a minimum number of bits that get decimated at each iteration, even if all information bits have a lower bias than the threshold. However, for the proposed algorithm that option is not available. Consequently we should use a small percentage of check nodes with degree 1 or equivalently we can initialize the information bits prior with a small value, at the first iteration, and remove it in the next iteration. In all simulation presented in this article the information bits prior was initialized with the value 0.1, and set to zero in the second iteration.

The procedure used to quantize a given source sequence is to run the BP algorithm with soft decimation, given by equations (15), (16) and (17), for a fixed ξ value and for a given number of iterations, and to decimate all information bits after.

The degree distribution pair $(\lambda(x), \rho(x))$ of the rate $1/2$ LDGM code used in all simulations shown in this article was:

$$\lambda(x) = x^6$$

$$\rho(x) = 0.275698x + 0.25537x^2 + 0.076598x^3 + 0.39233x^8$$

A. Distortion versus Compression Rate

Fig. 3 shows the obtained distortion as a function of the coding rate, for different source sequence lengths. For each code we choose ξ that achieve the lowest distortion. As can be seen from that figure the achieved distortion is very close to the theoretical limits. Even for a codeword length of 100 the achieved distortion is good, it is a little worse than the one obtained in [9] for a binary LDPC code of length 12000. We can also see, from that figure, that the distortion loss obtained by decreasing the number of iterations from 400 to 100 is small.

B. Distortion versus ξ

To attest the behavior of the distortion values has a stronger constraint is enforced we plot in Fig. 4 the distortion obtained by running the proposed algorithm, during 100 iterations, for a LDGM code with rate $1/2$ and for three codeword-lengths, 10^2, 10^4, 10^5. Remember that the parameter μ is inversely proportional to ξ. From Fig. 4 it is evident that has a stronger constraint is imposed the distortion value decrease but after the optimal ξ value it increases abruptly. Showing the existence of a threshold. As the codeword length increase the distortion value decreases even more abruptly, close to the optimal ξ value.

C. Distortion versus Codeword Length

Figure 5 presents the distortion values, as a function of the codeword length, for the rate one half LDGM code. For each code we choose ξ so that the average distortion is minimal. The ξ value is found by simulations. The number of iterations is fixed to 400 and the distortion was averaged over 100 trials, for codeword lengths higher than 1000, and averaged over 1000 trials otherwise. As can be observed, from figure 5, the average distortion obtained, has the codeword length increase, decrease exponentially and the gains obtained by the joint processing of a higher number of samples, from the interference sequence, gets smaller has the codeword length increase.

D. Computational Complexity

In this section we analyze the number of iterations needed for convergence, has the codeword length increase. To limit the impact of the graph cycles in the results we have considered high codeword length codes only, from $N = 10^4$ to $N = 10^6$. All these codes were generated randomly with only double edges removed. To analyze the number of iterations needed for convergence we have iterated our algorithm for 1000 iterations and averaged the corresponding obtained distortion over 1000 trials, for the rate $1/2$ code and for different codeword lengths. In Fig. 5 we plot the corresponding average

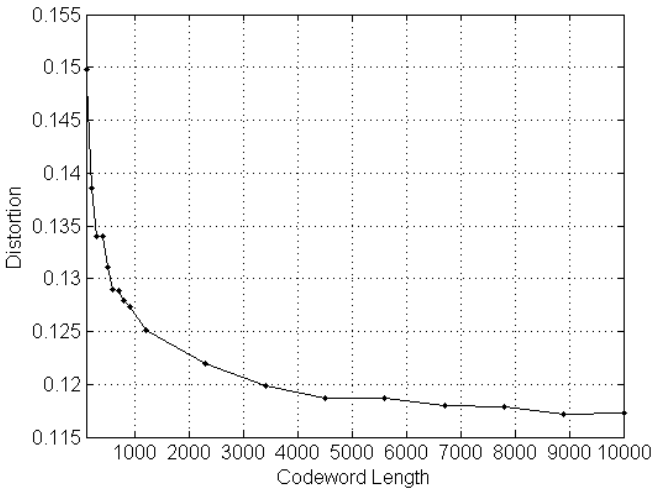

Fig. 5. Distortion values for various codeword lengths, from 10^2 to 10^4, for a LDGM code with R = 1/2 and for 400 iterations

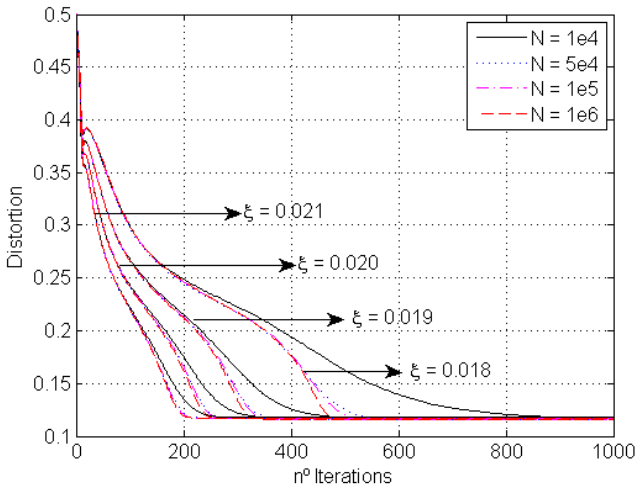

Fig. 6. Distortion evolution over the iteration number for various codeword lengths, for a code rate of 1/2.

distortion obtained for each iteration and for different ξ values. As can be seen from that figure the number of iterations needed for convergence decrease has the codeword length increase or has ξ increase. However the number of needed iterations seem to be saturating has the codeword length increase and they saturate faster for higher ξ values (weaker constraint/lower average distortion).

Even if one cannot extrapolate general conclusions from this numerical results, they indicate that the number of iterations needed for convergence tend to a constant value for very high codeword lengths and a constant ξ value. Supporting the $\mathcal{O}(N)$ computational complexity of the proposed algorithm.

V. Conclusion

In this paper we have proposed a new algorithm for lossy source coding, using LDGM codes. The main idea behind the proposed algorithm was to transform the decimation step, present in almost all previously proposed algorithms, into a soft-decimator and to include it in the BP algorithm. To do that we first have obtained an equivalent representation of the decimation step, with the help of an indicator function, and have included it into the BP update equations. After that we have proposed a new formulation for the aforementioned algorithm using a linear or 'soft' constraint function instead of a 'hard' constraint. The derived algorithm has linear complexity in the code block length. Simulation results indicate that close to the state-of-the art performance can be achieved, with reduced complexity.

Acknowledgment

This work has been performed in the framework of the FUTON-FP7-ICT-2007-215533, which is partially funded by the European Commission and PHOTON-PTDC-EEA-TEL-72890-2006 projects. D. Castanheira acknowledges support from the Fundação para a Ciência e a Tecnologia (FCT) of Portugal through a Doctoral Fellowship (SFRH/BD/41094/2007). The authors would also like to thank the three anonymous reviewers for their constructive comments and suggestions.

References

[1] M. Costa, "Writing on dirty paper (corresp.)," *Information Theory, IEEE Transactions on*, vol. 29, no. 3, pp. 439–441, May 1983.

[2] P. Moulin and J. O'Sullivan, "Information-theoretic analysis of information hiding," in *Information Theory, 2000. Proceedings. IEEE International Symposium on*, 2000, pp. 19–.

[3] R. Barron, B. Chen, and G. Wornell, "The duality between information embedding and source coding with side information and some applications," *Information Theory, IEEE Transactions on*, vol. 49, no. 5, pp. 1159–1180, May 2003.

[4] T. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *Information Theory, IEEE Transactions on*, vol. 47, no. 2, pp. 599–618, Feb 2001.

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

[6] E. Martinian and J. S. Yedidia, "Iterative quantization using codes on graphs," *CoRR*, vol. cs.IT/0408008, 2004.

[7] M. J. Wainwright and E. N. Maneva, "Lossy source encoding via message-passing and decimation over generalized codewords of ldgm codes," *CoRR*, vol. abs/cs/0508068, 2005.

[8] T. Filler and J. J. Fridrich, "Binary quantization using belief propagation with decimation over factor graphs of ldgm codes," *CoRR*, vol. abs/0710.0192, 2007.

[9] A. Braunstein, F. Kayhan, and R. Zecchina, "Efficient ldpc codes over gf(q) for lossy data compression," *CoRR*, vol. abs/0901.4467, 2009.

[10] T. M. Cover and J. A. Thomas, *Elements of information theory*. New York, NY, USA: Wiley-Interscience, 1991.

[11] J. Garcia-Frias and W. Zhong, "Approaching shannon performance by iterative decoding of linear codes with low-density generator matrix," *Communications Letters, IEEE*, vol. 7, no. 6, pp. 266–268, June 2003.

[12] A. Braunstein, F. Kayhan, G. Montorsi, and R. Zecchina, "Encoding for the blackwell channel with reinforced belief propagation," in *Information Theory, 2007. ISIT 2007. IEEE International Symposium on*, 24-29 2007, pp. 1891 –1895.

[13] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.