The Turbo Principle in Joint Source-Channel Coding

Joachim Hagenauer and Norbert Görtz
Institute for Communications Engineering (LNT)
Munich University of Technology (TUM)
80290 Munich, Germany
{hagenauer,norbert.goertz}@ei.tum.de

Abstract — **The turbo principle (iterative decoding between component decoders) is a general scheme, which we apply to joint source-channel decoding. As a realistic (e.g. speech parameter coding) example we discuss joint source-channel decoding for auto-correlated continuous-amplitude source samples. At the transmitter the source samples are quantized and their indexes are appropriately mapped onto bitvectors. Afterwards the bits are interleaved and channel-encoded; an AWGN channel is assumed for transmission. The auto-correlations of the source samples act as implicit outer channel codes that are serially concatenated with the inner explicit channel code. Thus, by applying the turbo principle, we can perform iterative decoding at the receiver. As an example, we will show that with a proper bit mapping for a 5-bit quantizer iterative source-channel decoding saves up to 2 dB in channel SNR or 8 dB in source SNR for an auto-correlated Gaussian source.**

I. Introduction and System Model

Figure 1 shows our model of a transmission system. A set of input source signals[1] has to be transmitted at each time index k; to simplify the notation, we will only consider one of the inputs, the samples X_k, which are quantized ("source-encoded") by the bitvector

$$I_k \doteq \{I_{k,1}, ..., I_{k,n}, ..., I_{k,N}\} \in \mathcal{I}\,, \qquad (1)$$

with $I_{k,n} \in \{0,1\}$ and

$$\mathcal{I} \doteq \{0,1\}^N \qquad (2)$$

denoting the set of all possible N-bit vectors. The quantizer reproduction value (or vector) corresponding to the bitvector I_k is denoted by $\hat{x}(I_k)$. Placed together with all parallel data in the bitvector U_k, the bitvectors are bit-interleaved and jointly channel-encoded; the N_v-bit channel codeword $V_k = \{V_{k,n}, n = 1, ..., N_v\}$ is transmitted over an AWGN-channel. Since we assume coherently detected binary modulation (phase-shift keying), the conditional pdf of the received value $y_{k,n}$ at the channel output, given that the code bit $v_{k,n} \in \{0,1\}$ has been transmitted, is given by

$$p_c(y_{k,n}|v_{k,n}) = \frac{e^{-\frac{1}{2\sigma_n^2}(y_{k,n}-(1-2\cdot v_{k,n}))^2}}{\sqrt{2\pi}\sigma_n} \qquad (3)$$

with the variance $\sigma_n^2 = \frac{N_0}{2E_s}$. E_s is the energy that is used to transmit each channel-code bit and N_0 is the one-sided power spectral density of the channel noise. The joint conditional pdf $p_c(y_k|v_k)$ for a channel word $y_k \in I\!R^{N_v}$ to be received, given that the codeword $v_k \in \{0,1\}^{N_v}$ is transmitted, is the product of (3) over all code-bits, since the channel noise is statistically independent.

[1] Capital letters are used to denote random variables, small letters are used for their realizations.

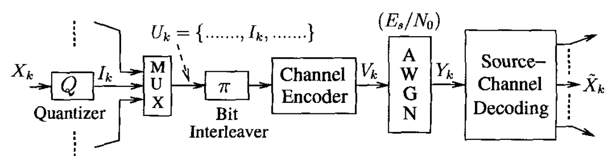

Figure 1: System Model

If the samples X_k are autocorrelated, adjacent bitvectors I_{k-1}, I_k show dependencies. They are modeled by a first-order stationary Markov-process, which is described by the transition probabilities $P(I_k|I_{k-1})$. We will assume that these probabilities and the unconditional probability-distributions of the bitvectors are known. Moreover, we will assume that the bitvectors I_k are independent[2] of all other data, which is transmitted in parallel by the bitvector U_k.

II. Joint Source-Channel Decoding (JSCD)

For the system model stated above, the goal is to minimize the distortion of the decoder output signal $\tilde{x}_k$ due to the channel noise, i.e., we want to perform joint source-channel *de*coding (JSCD) for a *fixed* transmitter. The mathematical formulation of the optimization criterion is given by the conditional expectation of the mean-squared error:

$$D \doteq \mathrm{E}_{I_k|\boldsymbol{y_k}}\left\{\left\|\tilde{x}_k - \hat{x}(I_k)\right\|_2^2 \,\middle|\, \boldsymbol{y_k}\right\}. \qquad (4)$$

In (4), $\hat{x}(I_k)$ is the quantizer reproduction value corresponding to the bitvector I_k, which is used by the source encoder to quantize X_k, and

$$\boldsymbol{y_k} \doteq \{y_0, y_1, ..., y_k\} \qquad (5)$$

is the set of channel output words which were received up to the current time k.

The minimization of (4) with respect to the decoder output signal $\tilde{x}_k$ results in the minimum mean-square estimator

$$\tilde{x}_k = \mathrm{E}_{I_k|\boldsymbol{y_k}}\{\hat{x}(I_k)\} = \sum_{I_k \in \mathcal{I}} \hat{x}(I_k) \cdot P(I_k \mid \boldsymbol{y_k}). \qquad (6)$$

[2] This assumption greatly simplifies the analysis and the realization of iterative source-channel decoding, and it is at least approximately fulfilled in many practically relevant situations.

By use of the Bayes-rule, we can show that the *bitvector a-posteriori probabilities* (APPs) are given by

$$P(I_k \mid \boldsymbol{y_k}) = B_k \cdot P(I_k \mid \boldsymbol{y_{k-1}}) \cdot p(y_k \mid I_k, \boldsymbol{y_{k-1}}) \quad (7)$$

where $P(I_k \mid \boldsymbol{y_{k-1}})$ is the *bitvector* a-priori probability, which is computed only from channel words that have been received in the past, i.e., the information carried by the currently received channel-word y_k is not used. The factor $B_k \doteq p(\boldsymbol{y_{k-1}})/p(\boldsymbol{y_k})$ is a normalizing constant, that makes the left-hand side of (7) a true probability that sums up to one over all possible bitvectors I_k. Thus, the pdfs $p(\boldsymbol{y_{k-1}})$ and $p(\boldsymbol{y_k})$ do not have to be computed explicitly.

Since $P(I_k \mid I_{k-1}, \boldsymbol{y_{k-1}}) = P(I_k \mid I_{k-1})$, it is straightforward to show that the *bitvector* a-priori probabilities are given by

$$\begin{aligned} P(I_k \mid \boldsymbol{y_{k-1}}) &= \sum_{I_{k-1} \in \mathcal{I}} P(I_k, I_{k-1} \mid \boldsymbol{y_{k-1}}) \\ &= \sum_{I_{k-1} \in \mathcal{I}} \underbrace{P(I_k | I_{k-1})}_{\text{Markov-model}} \cdot \underbrace{P(I_{k-1} | \boldsymbol{y_{k-1}})}_{\text{old APP}}, \quad (8) \end{aligned}$$

i.e., they are computed from the "old" *bitvector* APPs (7) at time $k-1$ and from the transition probabilities of the Markov model. For initialization at time $k=0$ the unconditional probability distribution of the bitvectors is used instead of the "old" APPs.

The term $p(y_k \mid I_k, \boldsymbol{y_{k-1}})$ on the right-hand side of (7) is, however, hard to compute analytically [1], because it is given by a sum over *all* possible data-words U_k with a specific bitvector I_k.

III. Iterative Source-Channel Decoding (ISCD)

Even if the number of jointly channel-encoded data bits (size of U_k) is only moderate, the optimal decoding algorithm stated in section II is practically infeasible due to the tremendous complexity in the computation of $p(y_k \mid I_k, \boldsymbol{y_{k-1}})$. Therefore, a less complex way to compute at least a good approximation is required; this idea will lead to iterative source-channel decoding [2]. As a first step towards the solution we write

$$p(y_k \mid I_k, \boldsymbol{y_{k-1}}) = \frac{p(y_k, I_k, \boldsymbol{y_{k-1}})}{p(I_k, \boldsymbol{y_{k-1}})} . \quad (9)$$

Now, in the numerator and in the denominator, the *bitvector* probability densities are approximated by the product over the corresponding *bit* probability densities:

$$p(y_k \mid I_k, \boldsymbol{y_{k-1}}) \approx \frac{\prod_{n=1}^{N} p(y_k, I_{k,n}, \boldsymbol{y_{k-1}})}{\prod_{n=1}^{N} p(I_{k,n}, \boldsymbol{y_{k-1}})} , \quad (10)$$

with the bits $I_{k,n} \in \{0, 1\}$. If we insert (10) into (7) we obtain

$$P(I_k \mid \boldsymbol{y_k}) \approx P(I_k \mid \boldsymbol{y_{k-1}}) \cdot \prod_{n=1}^{N} \frac{P(I_{k,n} \mid \boldsymbol{y_k})}{P(I_{k,n} \mid \boldsymbol{y_{k-1}})} . \quad (11)$$

The *bit* a-posteriori probabilities $P(I_{k,n} \mid \boldsymbol{y_k})$ can be efficiently computed by the symbol-by-symbol APP algorithm in [3] (BCJR algorithm), if a *binary* convolutional channel code with a small number of states is used. It should be noticed that all the received channel words $\boldsymbol{y_k}$ up to the current time k are used for the computation of the bit APPs, because the bit-based a-priori information

$$P(I_{k,n} \mid \boldsymbol{y_{k-1}}) = \sum_{I_k \in \mathcal{I} | I_{k,n}} P(I_k \mid \boldsymbol{y_{k-1}}) . \quad (12)$$

(for a specific bit $I_{k,n} \in \{0, 1\}$) that is used by the APP algorithm is derived from the time-correlations (described by (8)) of the bitvectors at the quantizer output.

We can interpret the fraction in (11) as the extrinsic information $P_e^{(C)}(I_{k,n})$ [4] that we get from the channel decoder and write

$$P(I_k \mid \boldsymbol{y_k}) \approx P(I_k \mid \boldsymbol{y_{k-1}}) \cdot \left[\prod_{n=1}^{N} P_e^{(C)}(I_{k,n}) \right] . \quad (13)$$

Note that we have introduced the superscript "(C)" to indicate that $P_e^{(C)}(I_{k,n})$ is the extrinsic information produced by the channel decoder. Formula (13) is strongly similar to the one that is used in the optimal-estimation algorithm [1], that has been introduced in [5] for soft-bit source decoding: instead of a pure channel-term that inserts the reliabilities of the received bits into the estimation, we have a modified channel-term in (13) (between the brackets) that includes the reliabilities of the received bits *and*, additionally, the information derived by the APP-algorithm from the channel-code.

In principle, we now could compute the mean square estimates $\tilde{x}_k$ for transmitted signals by (6) using the results from (13), but the *bitvector* APPs are only approximations of the optimal values, since the *bit* a-priori informations that were used for channel decoding did not contain the mutual dependencies of the bits within the bitvectors: they were removed by the summation in (12).

The idea how to improve the accuracy of the *bitvector* APPs is taken over from iterative decoding of turbo codes [6]: from the intermediate results for the *bitvector* APPs (13), new *bit* APPs are computed by

$$P^{(S)}(I_{k,n} \mid \boldsymbol{y_k}) = \sum_{I_k \in \mathcal{I} | I_{k,n}} P(I_k \mid \boldsymbol{y_k}) . \quad (14)$$

The superscript "(S)" was introduced, since (14) is computed after source decoding (which included the results of the previous channel-decoding step). Now, we can derive new bit extrinsic information, this time from the source decoder, by

$$P_e^{(S)}(I_{k,n}) \doteq \frac{P^{(S)}(I_{k,n} \mid \boldsymbol{y_k})}{P_e^{(C)}(I_{k,n})} , \quad (15)$$

where in the numerator the result of (14) is inserted.

The extrinsic information from the last run of the channel decoder is removed by the denominator in (15), since we do not want to loop back information to the channel decoder that it has produced itself in the previous iteration. The extrinsic information computed by (15) is used as the new a-priori information for the second and further runs of the channel decoder.

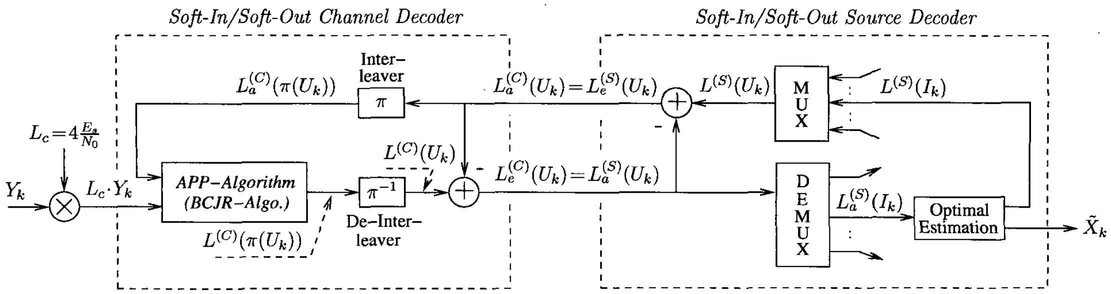

Figure 2: Iterative Source-Channel Decoding according to the Turbo Principle

Summary of iterative source-channel decoding: (ISCD)

1. At each time k, compute the initial *bitvector* a-priori probabilities by (8).
2. Use the results from step 1 in (12) to compute the initial *bit* a-priori information for the APP channel decoder.
3. Perform APP channel decoding.
4. Perform source decoding by inserting the extrinsic *bit* information from APP channel decoding into (13) to compute new (temporary) *bitvector* APPs.
5. If this is the last iteration proceed with step 8, otherwise continue with step 6.
6. Use the *bitvector* APPs of step 4 in (14), (15) to compute extrinsic *bit* information from the source redundancies.
7. Set the extrinsic *bit* information from step 6 equal to the new *bit* a-priori information for the APP channel decoder in the next iteration; proceed with step 3.
8. Estimate the receiver output signals by (6) using the *bitvector* APPs from step 4

The iterative source-channel decoder is depicted in Figure 2. It consists of two constituent decoders: the APP-algorithm for channel decoding and the Optimal-Estimation algorithm for source-decoding described above. The information that is passed between the constituent decoders are not probabilities, but log-likelihood-ratios (L-values, [4]) of the data bits, which are directly related to them. The L-values in Figure 2 are arranged in vectors for sake of clarity.

The computation of the bitvector APPs by (13) requires the bit-probabilities $P_e^{(C)}(I_{k,n})$. The latter can be computed from the output L-values [4]

$$L_e^{(C)}(I_{k,n}) \doteq \log \frac{P_e^{(C)}(I_{k,n}=0)}{P_e^{(C)}(I_{k,n}=1)} \tag{16}$$

of the APP channel decoder by inversion of the definition (16):

$$P_e^{(C)}(I_{k,n}) = \frac{e^{L_e^{(C)}(I_{k,n})}}{1+e^{L_e^{(C)}(I_{k,n})}} \cdot e^{-L_e^{(C)}(I_{k,n})\cdot I_{k,n}} \,. \tag{17}$$

Note that in (17) the L-values $L_e^{(C)}(I_{k,n})$ are fixed real numbers, while the bit $I_{k,n} \in \{0,1\}$ denotes a particular "value" that is selected at the left-hand side. Since the Optimal-Estimation algorithm computes the product over all these probabilities for the bitvector, this operation can be simplified by inserting (17) into (13) and by turning the product over the exponential functions into summations in the exponents:

$$P(I_k|\boldsymbol{y_k}) \approx A_k \cdot P(I_k|\boldsymbol{y_{k-1}}) \cdot \exp\Big(-\sum_{n=1}^{N} L_e^{(C)}(I_{k,n}) \cdot I_{k,n}\Big). \tag{18}$$

The normalizing constant A_k does not depend on the variable $I_{k,n}$. Thus, the L-values from the APP channel decoder can be integrated into the Optimal-Estimation algorithm for APP source decoding without converting the individual L-values back to bit probabilities if (18) is used instead of (13). This is a simplification that also has strong numerical advantages.

The computation of new bit APPs within the iterations is still carried out by (14), but the derivation of the extrinsic L-values $L_e^{(S)}(I_{k,n})$, that are issued by APP source decoder, can be simplified, since (15) requires a division which is turned into a simple subtraction in the L-value-domain:

$$L_e^{(S)}(I_{k,n}) = L^{(S)}(I_{k,n}) - L_e^{(C)}(I_{k,n}) \,. \tag{19}$$

Thus, in the ISCD algorithm the L-values $L_e^{(C)}(I_{k,n})$ from the APP channel decoder are used and the probabilities $P_e^{(C)}(I_{k,n})$ are not required.

IV. Quantizer Bit Mappings

Due to the low-pass correlation which we assume for the input[3], the value of the sample x_k will be close to x_{k-1}. Thus, if e.g. the quantizer output (see Fig. 3) at time $k-1$ is $\hat{x}_1$, then the next quantizer output at time k will be $\hat{x}_0, \hat{x}_1$, or $\hat{x}_2$ with high probability, while the probability for, say, $\hat{x}_7$ is small.

If the channel code is strong enough so that the extrinsic L-values at the APP channel decoder output have large magnitudes, we can idealize this situation by assuming, that the a-priori information for the source decoder is perfect; within the iterative decoding scheme this simply means that the APP source decoder tries to generate extrinsic information for a particular data bit, while it exactly knows all other bits. The situation is illustrated in Figure 3. We assume that a 3-bit scalar quantizer with the reproduction levels $\hat{x}_0, ..., \hat{x}_7$ is used for encoding of correlated source samples. The quantizer levels are given three different bit mappings[4]: natural binary, Gray, and optimized for ISCD. As an example, we consider the case

[3]The concept can be easily extended to high-pass correlations.

[4]In [7] some bit mappings are analyzed for iterative source-channel decoding but no attempt is made to optimize them.

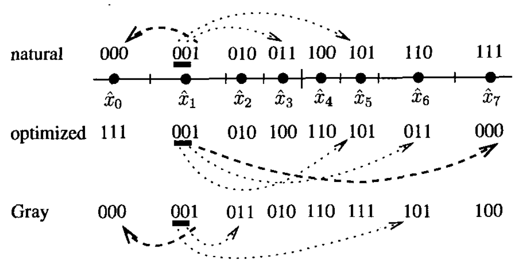

Figure 3: Bit Mappings for a 3-bit Quantizer to be used in Iterative Source-Channel Decoding

that the bitvector of the reproduction value $\hat{x}_1$ has been transmitted and that the two leftmost bits are known (both are zero), due to the a-priori information from the channel decoder. We now try to generate extrinsic information for the rightmost bit.

If we use the natural or the Gray mapping and flip the rightmost bit, we end up with quantizer level $\hat{x}_0$ instead of $\hat{x}_1$. Since $\hat{x}_0$ and $\hat{x}_1$ are neighbors in the source signal space we cannot decide with low error probability whether the rightmost bit is "one" or "zero," because from the source correlations we know that both $\hat{x}_0$ *and* $\hat{x}_1$ are highly probable.

The situation is totally different if we use the optimized mapping: since we jump to quantizer level $\hat{x}_7$ (which is highly improbable) if we flip the rightmost bit, we can take a sure decision for the bit in favor of "one". Thus, the magnitude of the corresponding extrinsic information will be large and it will help the channel decoder in the next iteration.

The example above suggests the concept for a good choice of the bit mapping: we have to allocate bitvectors to the quantizer reproduction levels such that if we flip one of the bits, each pair of reproduction values has a *large* Euclidean distance in the source signal space. We used a numerical approach, based on "binary switching," to optimize the mappings: we formulated the total source-signal distortion in case of 1-bit errors as a sum of individual distortions for the reproduction levels. Then we *maximized* the total distortion stepwise by switching the bit mapping of the reproduction level producing the *lowest* individual distortion with the bit mapping of another reproduction level. We repeated this process until no further switch was possible that increased the total distortion.

V. Simulations Results

We correlated independent Gaussian random samples by a first-order recursive low-pass filter (coefficient $a = 0.9$) in order to generate the input signals x_k. As source encoders, we used 5-bit Lloyd-Max scalar quantizers. This way we generated 50 mutually independent bitvectors, all transmitted at time index k. The bits were scrambled by a random-interleaver and afterwards they were commonly channel-encoded by a rate-1/2 recursive systematic convolutional code (RSC-code, [6]) with memory 4, which we terminated after each block of 50 bitvectors (250 bits). The codewords were transmitted over the AWGN-channel. At the decoder iterative source-channel decoding was performed.

The results are depicted in Figure 4. The performance of ISCD with the optimized bit mapping is much better than for the other mappings as long as $E_b/N_0 > 0$ dB (which is the realistic operating range). The full gain in source SNR is reached

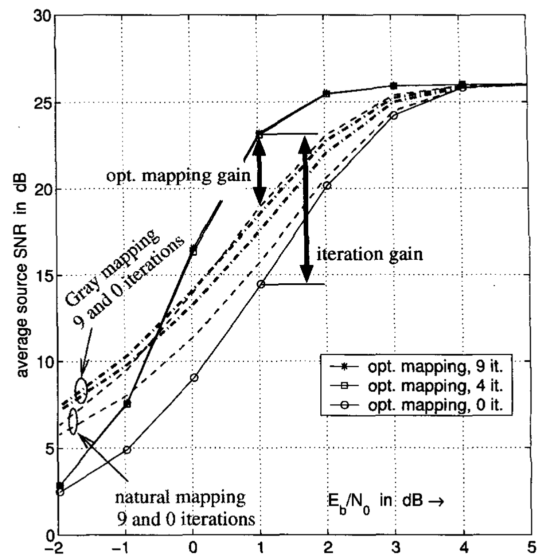

Figure 4: Performance of Iterative Source-Channel Decoding for various 5-Bit Mappings; Correlated ($a = 0.9$) Gaussian Source; Memory-4 Rate-1/2 Convolutional Code

after four iterations; it takes, however, at least one iteration before the optimized mapping works better than the other ones. For very low channel SNR the Gray mapping works best, because the channel decoder cannot generate reliable a-priori information. The natural mapping is a bad choice in any case.

VI. Conclusions

We have demonstrated that strong quality gains are achievable by the application of the turbo principle in joint source-channel decoding. We have shown that the bit mapping of the quantizers is important for the performance. Moreover, we have optimized the bit mapping of the quantizers for application in iterative source-channel decoding and we again obtained strong quality improvements.

References

[1] N. Görtz, "Joint source-channel decoding by channel-coded optimal estimation," in *Proceedings of the 3rd ITG Conference Source and Channel Coding (VDE Verlag)*, pp. 267–272, Jan. 2000.

[2] N. Görtz, "A generalized framework for iterative source-channel decoding," *Annals of Telecommunications, Special issue on "Turbo Codes: a wide-spreading Technique"*, pp. 435–446, Jul./Aug. 2001.

[3] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. IT-20, pp. 284–287, Mar. 1974.

[4] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Transactions on Information Theory*, vol. 42, pp. 429–445, Mar. 1996.

[5] T. Fingscheidt and P. Vary, "Robust speech decoding: A universal approach to bit error concealment," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 3, pp. 1667–1670, Apr. 1997.

[6] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Transactions on Communications*, vol. 44, pp. 1261–1271, Oct. 1996.

[7] M. Adrat, P. Vary, and J. Spittka, "Iterative source-channel decoder using extrinsic information from softbit source decoding," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. IV, pp. 2653–2656, May 2001.