Optimization of Bit Mappings for Iterative Source-Channel Decoding

Norbert Görtz

Institute for Communications Engineering (LNT),
Munich University of Technology (TUM), 80290 Munich, Germany
Phone: (+49) 89 289 23494, Fax: (+49) 89 289 23490
E-mail: norbert.goertz@ei.tum.de

Abstract: *The transmission of continuous-valued, auto-correlated source samples is considered. The samples are quantized and the resulting indexes are mapped onto bitvectors; the bits are interleaved and channel-encoded. This coding scheme serially concatenates the auto-correlations of the source samples and the inner explicit channel code. Thus, iterative source-channel decoding based on the turbo principle can be used. As a novelty, an algorithm is proposed for the optimization of the quantizer bit mappings for iterative decoding. The simulations show strong gains due to the optimization.*

Keywords: Joint Source-Channel Coding, Iterative Decoding, Optimal Bit-Mapping, Binary Switching

1. Introduction and System Model

Figure 1 shows our model of a transmission system. A set of mutually independent but auto-correlated input source signals[1] is 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 by the bitvector

$$b_k \doteq \{b_{k,1}, ..., b_{k,n}, ..., b_{k,N}\} \in \mathcal{B}\,, \qquad (1)$$

with $b_{k,n} \in \{0,1\}$ and $\mathcal{B} \doteq \{0,1\}^N$ denoting the set of all possible N-bit vectors. The quantizer reproduction level (possibly a vector) corresponding to the bitvector b_k is denoted by $\hat{x}(b_k)$. Placed together with all parallel data in the bitvector u_k, the bitvectors b_k are bit-interleaved (by a new random interleaver at each time k) 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[2] of the channel output value $\tilde{v}_{k,n}$, given that the code bit $v_{k,n} \in \{0,1\}$ has been transmitted, is given by

$$p_c(\tilde{v}_{k,n}|v_{k,n}) = \frac{1}{\sqrt{2\pi}\sigma_n} e^{-\frac{1}{2\sigma_n^2}(\tilde{v}_{k,n}-(1-2\cdot v_{k,n}))^2}\,, \quad (2)$$

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 $N_0/2$ is the power spectral density of the channel noise. The joint conditional pdf $p_c(\tilde{v}_k|v_k)$ for a channel word $\tilde{v}_k \in \mathbb{R}^{N_v}$ to be received, given that the codeword $v_k \in \{0,1\}^{N_v}$ is transmitted, is the product of (2) over all code-bits, since the channel noise is statistically independent.

[1] Such a setup represents a realistic model for the coded transmission of, e.g., speech, audio, or image signals, where the actual source signal is decomposed into parameters—the source signals in our system model—which are quantized and transmitted over a noisy channel.

[2] For brevity we will only use the realizations of the random variables—denoted by small letters—in all formulas, as long as there is no risk of confusion. The random variables are denoted by capital letters; they are introduced if required.

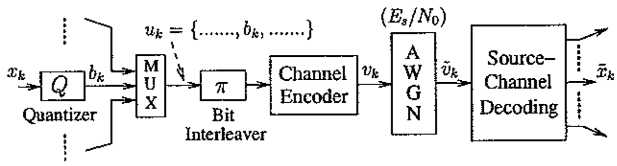

Figure 1: System model

As the samples x_k are correlated, adjacent bitvectors b_{k-1}, b_k show dependencies. They are modeled by a first-order stationary Markov-process, which is described by the (known) transition probabilities $P(b_k|b_{k-1})$.

2. Joint Source-Channel Decoding

The goal is to minimize the distortion of the decoder output $\tilde{x}_k$ due to channel noise, i.e., we want to perform joint source-channel *de*coding (JSCD) for a fixed transmitter. Hence, we minimize the conditional expectation

$$\mathrm{E}_{B_k|\tilde{V}_k}\left\{\left\|\tilde{x}_k - \hat{x}(B_k)\right\|_2^2 \,\middle|\, \tilde{\boldsymbol{v}}_k\right\} \qquad (3)$$

with respect to $\tilde{x}_k$, which results in the minimum mean-square estimator

$$\tilde{x}_k = \mathrm{E}_{B_k|\tilde{V}_k}\{\hat{x}(B_k)|\tilde{\boldsymbol{v}}_k\} = \sum_{b_k \in \mathcal{B}} \hat{x}(b_k)\cdot P(b_k|\tilde{\boldsymbol{v}}_k), \quad (4)$$

where $\tilde{\boldsymbol{v}}_k \doteq \{\tilde{v}_0, \tilde{v}_1, ..., \tilde{v}_k\}$ is the set of all channel-outputs up to the current time k.

By use of the Bayes-rule, we can show that the *bitvector a-posteriori probabilities* (APPs) in (4) are given by

$$P(b_k\,|\,\tilde{\boldsymbol{v}}_k) = A_k \cdot P(b_k\,|\,\tilde{\boldsymbol{v}}_{k-1}) \cdot p(\tilde{v}_k\,|\,b_k, \tilde{\boldsymbol{v}}_{k-1}), \quad (5)$$

where $P(b_k \mid \tilde{\boldsymbol{v}}_{k-1})$ is the *bitvector* a-priori probability at time k; $A_k \doteq p(\tilde{\boldsymbol{v}}_{k-1})/p(\tilde{\boldsymbol{v}}_k)$ is a normalizing factor, that ensures that the left-hand side of (5) sums up to one over all bitvectors $b_k \in \mathcal{B}$. Thus, the pdfs $p(\tilde{\boldsymbol{v}}_{k-1}), p(\tilde{\boldsymbol{v}}_k)$ do not have to be known explicitly.

Since $P(b_k\,|\,b_{k-1}, \tilde{\boldsymbol{v}}_{k-1}) = P(b_k\,|\,b_{k-1})$, the *bitvector* a-priori probabilities are given by

$$P(b_k\,|\,\tilde{\boldsymbol{v}}_{k-1}) = \sum_{b_{k-1}\in\mathcal{B}} \underbrace{P(b_k|b_{k-1})}_{\text{Markov-model}} \cdot \underbrace{P(b_{k-1}|\tilde{\boldsymbol{v}}_{k-1})}_{\text{old APPs}}\,, \quad (6)$$

i.e., they are computed from the "old" *bitvector* APPs (5) at time $k-1$ and from the transition probabilities of the Markov model; the information carried by the currently received channel-word $\tilde{v}_k$ is not used in (6). For initialization at time $k=0$ the unconditional probability distribution of the bitvectors is used instead of the "old" APPs.

3. Iterative Source-Channel Decoder (ISCD)

Even if the number of jointly channel-encoded data bits (size of u_k) is only moderate, optimal decoding is practically infeasible due to the tremendous complexity in the computation of $p(\tilde{v}_k \mid b_k, \tilde{\mathbf{v}}_{k-1})$ in (5). Therefore, a less complex way to compute an approximation has been stated in [1]: if we write $p(\tilde{v}_k|b_k, \tilde{\mathbf{v}}_{k-1}) = p(\tilde{v}_k, b_k, \tilde{\mathbf{v}}_{k-1})/p(b_k, \tilde{\mathbf{v}}_{k-1})$ and replace the *bitvector* probability densities by the products over the corresponding *bit* probability densities, we obtain the approximation

$$P(b_k \mid \tilde{\mathbf{v}}_k) \approx P(b_k \mid \tilde{\mathbf{v}}_{k-1}) \cdot \prod_{n=1}^{N} \frac{P(b_{k,n} \mid \tilde{\mathbf{v}}_k)}{P(b_{k,n} \mid \tilde{\mathbf{v}}_{k-1})} \cdot \quad (7)$$

from (5). The *bit* a-posteriori probabilities $P(b_{k,n} \mid \tilde{\mathbf{v}}_k)$ can be efficiently computed by the symbol-by-symbol APP algorithm [2] for a *binary* convolutional channel code with a small number of states. Although the APP-algorithm only decodes the currently received channel word $\tilde{v}_k$, it still uses all the received channel words $\tilde{\mathbf{v}}_k$ up to the current time k, because its bit-based a-priori information

$$P(b_{k,n} \mid \tilde{\mathbf{v}}_{k-1}) = \sum_{b_k \in \mathcal{B}|b_{k,n}} P(b_k \mid \tilde{\mathbf{v}}_{k-1}) \,. \quad (8)$$

(for a specific bit $b_{k,n} \in \{0,1\}$), is derived from the time-correlations (6) of the bitvectors at the quantizer output.

We can interpret the fraction in (7) as the extrinsic information $P_e^{(C)}(b_{k,n})$ [2], [3] that we get from the channel decoder and write

$$P(b_k \mid \tilde{\mathbf{v}}_k) \approx P(b_k \mid \tilde{\mathbf{v}}_{k-1}) \cdot \left[\prod_{n=1}^{N} P_e^{(C)}(b_{k,n})\right] \cdot \quad (9)$$

Note that we have introduced the superscript "(C)" to indicate that $P_e^{(C)}(b_{k,n})$ is the extrinsic information produced by the APP channel-decoding algorithm.

In principle, we now could compute the mean-square estimates $\tilde{x}_k$ for transmitted signals by (4) using the *bitvector* APPs from (9), but the latter are only approximations of the optimal values, since the *bit* a-priori informations that were used for APP channel decoding did not contain the mutual dependencies of the bits within the bitvectors: they were removed by the summation in (8). Therefore, new *bit* APPs are computed from the intermediate results (9) by

$$P^{(S)}(b_{k,n} \mid \tilde{\mathbf{v}}_k) = \sum_{b_k \in \mathcal{B}|b_{k,n}} P(b_k \mid \tilde{\mathbf{v}}_k) \,. \quad (10)$$

The superscript "(S)" was introduced, as (10) is computed after source decoding. Now, we can derive new *bit* extrinsic information from the source decoder by

$$P_e^{(S)}(b_{k,n}) \doteq P^{(S)}(b_{k,n} \mid \tilde{\mathbf{v}}_k)/P_e^{(C)}(b_{k,n}) \,, \quad (11)$$

where in the numerator the result of (10) is inserted.

The extrinsic information from the last run of the channel decoder is removed by the denominator in (11), 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 (11) is used as the new a-priori information for the second and further runs of the APP channel decoder.

Summary of iterative source-channel decoding (ISCD):

1. At each time k, compute the initial *bitvector* a-priori probabilities by (6).
2. Use the results from step 1 in (8) to compute the initial *bit* a-priori information for APP channel decoding (step 3).
3. Perform APP channel decoding.
4. Perform source decoding by inserting the extrinsic *bit* information from APP channel decoding into (9) to compute new (temporary) *bitvector* APPs.
5. If this is the last of a given number of iterations, proceed with step 8, otherwise continue with step 6.
6. Use the *bitvector* APPs of step 4 in (10), (11) 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 (4) using the *bitvector* APPs from step 4

An efficient implementation of ISCD by use of L-values [2] has been stated in [1].

4. Quantizer Bit Mappings

4.1. Basic Considerations

Due to the low-pass correlation, which we assume for the input, the value of the sample x_k will be close to x_{k-1}. Thus, if the input at time $k-1$ is scalar quantized by, e.g., $\hat{x}_1$ (see Figure 2), the next quantized value 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.

For the transmission, the indexes i of the quantizer reproduction levels $\hat{x}_i$ are mapped to bitvectors b by

$$b = \gamma(i) \,, \quad i \in \{0,1,...,2^N-1\} \,, \quad b \in \mathcal{B} \,, \quad (12)$$

where the quantizer index i is simply the number of the row in the codebook in which the quantizer level is located. The inverse of (12), i.e. the mapping from a bitvector b to the corresponding quantizer index i, is denoted by $i = \gamma^{-1}(b)$. Some mappings are illustrated in Figure 2.

If the channel code is strong enough so that hard decisions for the data bits could be taken correctly with high probability, we can idealize this situation by assuming, that the (soft) 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

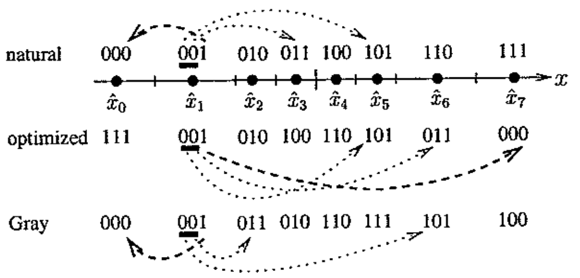

Figure 2: Bit mappings for an N-bit quantizer ($N = 3$)

knows all other bits exactly. The situation is illustrated in Figure 2: we assume that a 3-bit scalar quantizer with the reproduction levels $\hat{x}_0, \ldots, \hat{x}_7$ is used for encoding of low-pass correlated source samples x_k. The quantizer levels are given three different bit mappings: natural binary, Gray, and optimized for ISCD. As an example, we consider the case that the bitvector of the quantizer reproduction level $\hat{x}_1$ has been transmitted and that the two leftmost bits are known (both are zero in all mappings), due to the a-priori information from the channel decoder. We now try to generate extrinsic information for the rightmost bit from the source redundancies.

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 both $\hat{x}_0$ *and* $\hat{x}_1$ are highly probable due to the low-pass correlation of the source samples.

The situation is 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 in favor of "one." Thus, the extrinsic information generated by the APP source decoder is strong and it will aid the channel decoder in the next iteration.

The example above suggests the concept to optimize 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 levels has a *large* Euclidean distance in the source signal space. This still holds, if the source signal is high-pass correlated (with zero-mean and a symmetric probability density function): in this case the source sample x_k will be close to $-x_{k-1}$ with high probability. Thus, the goal is again to maximize the Euclidean distance between quantizer levels whose bit mappings differ in one bit position. This idea to optimize the bit mapping has only been mentioned in [4]; here the optimization algorithm is given in full detail.

4.2. Optimization by Binary Switching

As argued above, the optimization criterion, the total "cost," is the expected Euclidean distance

$$D \doteq \frac{1}{N} \sum_{z \in \mathcal{B}: w(z)=1} \mathrm{E}_I \left\{ \left\| \hat{x}_I - \hat{x}_{\gamma^{-1}(\gamma(I) \oplus z)} \right\|_2^2 \right\} \qquad (13)$$

between any two quantizer levels whose bit-mappings differ in exactly one bit position, averaged over all N possible bit positions. Thus in (13), the Hamming-weight $w(z)$ of the "error" bitvector z is fixed to one. The symbol "$\oplus$" denotes bitwise modulo-2 addition; γ is the bit mapping to be optimized. If we expand (13) we obtain

$$D = \sum_{i=0}^{2^N-1} C(i) \qquad (14)$$

for the total cost, with the individual costs

$$C(i) \doteq \frac{P(i)}{N} \sum_{z \in \mathcal{B}: w(z)=1} \left\| \hat{x}_i - \hat{x}_{\gamma^{-1}(\gamma(i) \oplus z)} \right\|_2^2 \qquad (15)$$

of the quantizer levels (also called "codevectors"); $P(i)$ denotes their known unconditional probabilities.

In principle it is possible to maximize the total cost D with respect to the mapping γ by the brute-force approach, but even if only $N = 5$ bits are used for quantization, the number of different mappings is extremely large ($2^N! \approx 2.6 \cdot 10^{35}$). Hence, we use a numerical approach with reasonable complexity that is based on the "binary switching algorithm" (BSA). It was proposed in [5] for a "Pseudo-Gray" reordering of the codevectors in vector quantizer codebooks in the context of channel-optimized quantization. For the BSA the quantizer levels are assumed to be given (e.g., by codebook training) and an initial bit mapping is either created randomly or it is derived from the ordering of the quantizer levels in the codebook: the row numbers of the quantizer levels are converted into the corresponding binary numbers (natural mapping).

The flowchart of the BSA, adapted to our case, is depicted in Figure 3. The bitvector corresponding to the codevector[3] with the *lowest* individual cost (15) is tried to be switched with the bitvector of another codevector, the "switch partner." The latter is selected such, that the increase of the total cost due to the switch is as large as possible. If no switch partner can be found for the quantizer level with the lowest cost (that means all possible switches result in a lower total cost), the bitvector of the quantizer level with the second-lowest cost will be tried to switch next. This process continues until a quantizer level from the list, sorted by increasing costs (15), is found that allows a switch of bitvectors that increases the total cost. After an accepted switch, the cost of each quantizer level and the total cost are recalculated, a new ordered list of quantizer levels is generated, and the algorithm continues as described above, until no further increase of the total cost is possible.

A weak upper bound for the number of switch-trials required to reach an optimum is given by the number $2^N!$ of different bit mappings [5]. In practice, however, the BSA converges much faster: a few hundred up to thousand accepted switches are usually required for $N < 10$.

[3] The term "codevector" is equivalently used for "quantizer level" to point out that the optimization scheme also works for vector quantizers.

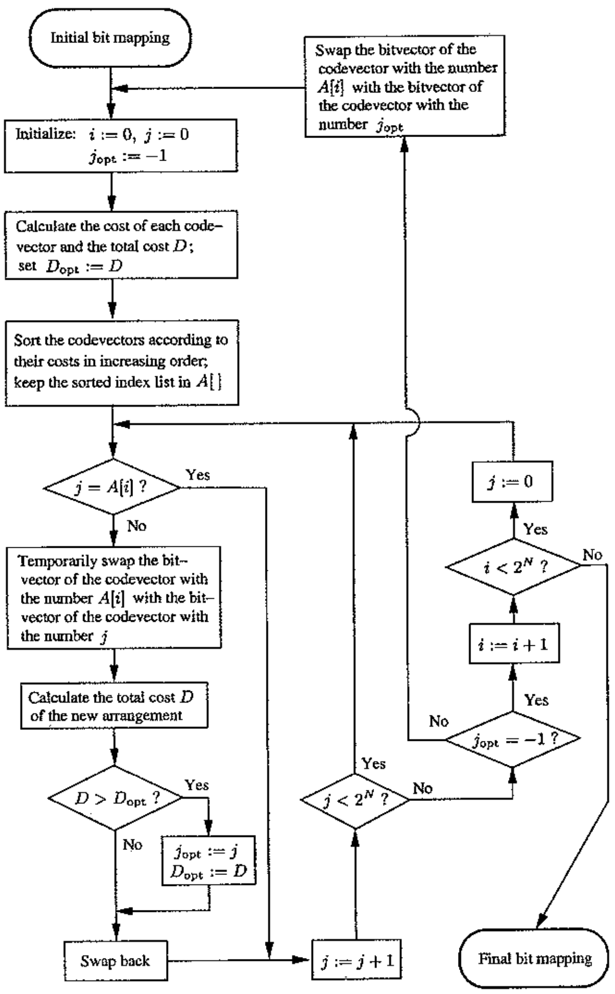

Figure 3: Flowchart of the Binary Switching Algorithm

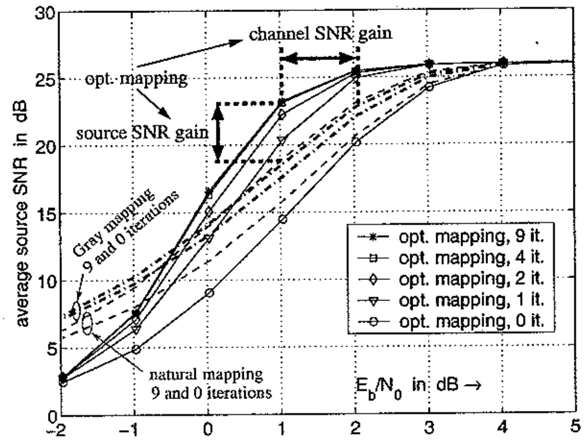

Figure 4: Performance of iterative source-channel decoding for several 5-bit mappings; correlated ($a = 0.9$) Gaussian source; terminated memory-4 convolutional code with rate $R = 250/508$; $E_b/N_0 = \frac{1}{R} E_s/N_0$; variance of the channel noise in the simulation: $\sigma_n^2 = N_0/(2E_s)$.

5. Simulation Results

We generated auto-correlated source signals by processing Gaussian samples with the low-pass filter $H(z) = \frac{z}{z-a}$, with $a = 0.9$. Fifty of these auto-correlated but mutually independent signals were source encoded, each by a 5-bit optimal scalar quantizer, and the quantizer output bitvectors were random-interleaved and channel encoded by a rate-1/2 recursive systematic convolutional code (RSC-code, [3]) with memory 4 (terminated after each block of 50 bitvectors, i.e., 250 bits). The codewords were transmitted over the AWGN-channel and iterative source-channel decoding was performed at the decoder.

The results are depicted in Figure 4. In the realistic operating range ($E_b/N_0 > 0$ dB), the performance of ISCD with the optimized bit mapping is much better than for the other mappings; compared to the Gray mapping the transmission power can be reduced by up to 1 dB for the same source SNR. The full gain is reached 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 should not be used in any case. For a larger number of jointly channel encoded bitvectors, the performance moderately increases due to the interleaver gain [1].

6. Conclusions

We have stated a new algorithm with realistic complexity for the optimization of the quantizer bit mappings for application in iterative source-channel decoding. The simulation results indicate that strong improvements in transmission quality are achieved by the optimization.

As the optimization method is quite general, it can also be used in other scenarios such as bit-interleaved coded modulation where iterative processing is applied and bit mappings shall be optimized.

REFERENCES

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

[2] 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.

[3] 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.

[4] J. Hagenauer and N. Görtz, "The turbo principle in joint source-channel coding," in *Proceedings of the IEEE Information Theory Workshop*, pp. 275–278, Apr. 2003.

[5] K. Zeger and A. Gersho, "Pseudo-Gray coding," *IEEE Transactions on Communications*, vol. 38, pp. 2147–2158, Dec. 1990.