# NEAR SHANNON LIMIT ERROR - CORRECTING CODING AND DECODING : TURBO-CODES (1)

Claude Berrou, Alain Glavieux and Punya Thitimajshima

Claude Berrou, Integrated Circuits for Telecommunication Laboratory

Alain Glavieux and Punya Thitimajshima, Digital Communication Laboratory

Ecole Nationale Supérieure des Télécommunications de Bretagne, France

(1) Patents N° 9105279 (France), N° 92460011.7 (Europe), N° 07/870,483 (USA)

***Abstract*** - This paper deals with a new class of convolutional codes called *Turbo-codes*, whose performances in terms of Bit Error Rate (BER) are close to the SHANNON limit. The *Turbo-Code* encoder is built using a parallel concatenation of two Recursive Systematic Convolutional codes and the associated decoder, using a feedback decoding rule, is implemented as $P$ pipelined identical elementary decoders.

## I - INTRODUCTION

Consider a binary rate $R$=1/2 convolutional encoder with constraint length $K$ and memory $M$=$K$-1. The input to the encoder at time $k$ is a bit $d_k$ and the corresponding codeword $C_k$ is the binary couple $(X_k, Y_k)$ with

$$X_k = \sum_{i=0}^{K-1} g_{1i} d_{k-i} \quad mod.2 \quad g_{1i} = 0,1 \qquad (1a)$$

$$Y_k = \sum_{i=0}^{K-1} g_{2i} d_{k-i} \quad mod.2 \quad g_{2i} = 0,1 \qquad (1b)$$

where $G_1$: $\{g_{1i}\}$, $G_2$: $\{g_{2i}\}$ are the two encoder generators, generally expressed in octal form.

It is well known, that the BER of a classical Non Systematic Convolutional (NSC) code is lower than that of a classical Systematic code with the same memory $M$ at large SNR. At low SNR, it is in general the other way round. The new class of Recursive Systematic Convolutional (RSC) codes, proposed in this paper, can be better than the best NSC code at any SNR for high code rates.

A binary rate $R$=1/2 RSC code is obtained from a NSC code by using a feedback loop and setting one of the two outputs $X_k$ or $Y_k$ equal to the input bit $d_k$. For an RSC code, the shift register (memory) input is no longer the bit $d_k$ but is a new binary variable $a_k$. If $X_k$=$d_k$ (respectively $Y_k$=$d_k$), the output $Y_k$ (resp. $X_k$) is equal to equation (1$b$) (resp. 1$a$) by substituting $a_k$ for $d_k$ and the variable $a_k$ is recursively calculated as

$$a_k = d_k + \sum_{i=1}^{K-1} \gamma_i a_{k-i} \qquad mod.2 \qquad (2)$$

where $\gamma_i$ is respectively equal to $g_{1i}$ if $X_k$=$d_k$ and to $g_{2i}$ if $Y_k$=$d_k$. Equation (2) can be rewritten as

$$d_k = \sum_{i=0}^{K-1} \gamma_i a_{k-i} \qquad mod.2. \qquad (3)$$

One RSC encoder with memory $M$=4 obtained from an NSC encoder defined by generators $G_1$=37, $G_2$=21 is depicted in Fig.1.

Generally, we assume that the input bit $d_k$ takes values 0 or 1 with the same probability. From equation (2), we can show that variable $a_k$ exhibits the same statistical property

$$P_r\{a_k = 0 / a_1 = \varepsilon_1, .. a_{k-1} = \varepsilon_{k-1}\} = P_r\{d_k = \varepsilon\} = 1/2 \quad (4)$$

with ε is equal to

$$\varepsilon = \sum_{I=1}^{K-1} \gamma_i \varepsilon_i \quad mod.2 \quad \varepsilon = 0,1. \quad (5)$$

Thus the trellis structure is identical for the RSC code and the NSC code and these two codes have the same free distance $d_f$. However, the two output sequences $\{X_k\}$ and $\{Y_k\}$ do not correspond to the same input sequence $\{d_k\}$ for RSC and NSC codes. This is the main difference between the two codes.

When punctured code is considered, some output bits $X_k$ or $Y_k$ are deleted according to a chosen puncturing pattern defined by a matrix $P$. For instance, starting from a rate $R$=1/2 code, the matrix $P$ of rate 2/3 punctured code is

$$P = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

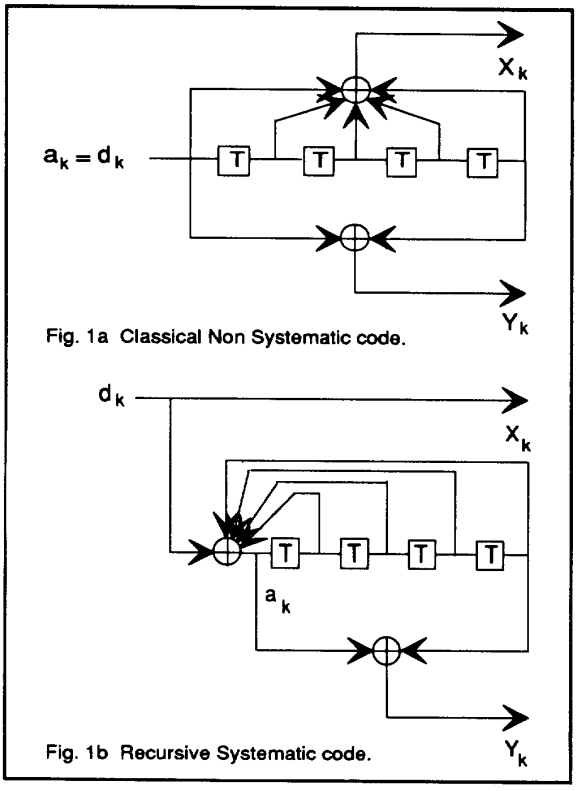


Fig. 1a Classical Non Systematic code.




Fig. 1b Recursive Systematic code.



0-7803-0950-2/93/$3.00©1993IEEE

## II - PARALLEL CONCATENATION OF RSC CODES

With RSC codes, a new concatenation scheme, called parallel concatenation can be used. In Fig. 2, an example of two identical RSC codes with parallel concatenation is shown. Both elementary encoder ($C_1$ and $C_2$) inputs use the same bit $d_k$ but according to a different sequence due to the presence of an interleaver. For an input bit sequence $\{d_k\}$, encoder outputs $X_k$ and $Y_k$ at time $k$ are respectively equal to $d_k$ (systematic encoder) and to encoder $C_1$ output $Y_{1k}$, or to encoder $C_2$ output $Y_{2k}$. If the coded outputs ($Y_{1k}$, $Y_{2k}$) of encoders $C_1$ and $C_2$ are used respectively $n_1$ times and $n_2$ times and so on, the encoder $C_1$ rate $R_1$ and encoder $C_2$ rate $R_2$ are equal to

$$R_1 = \frac{n_1 + n_2}{2n_1 + n_2} \qquad R_2 = \frac{n_1 + n_2}{2n_2 + n_1}. \qquad (6)$$

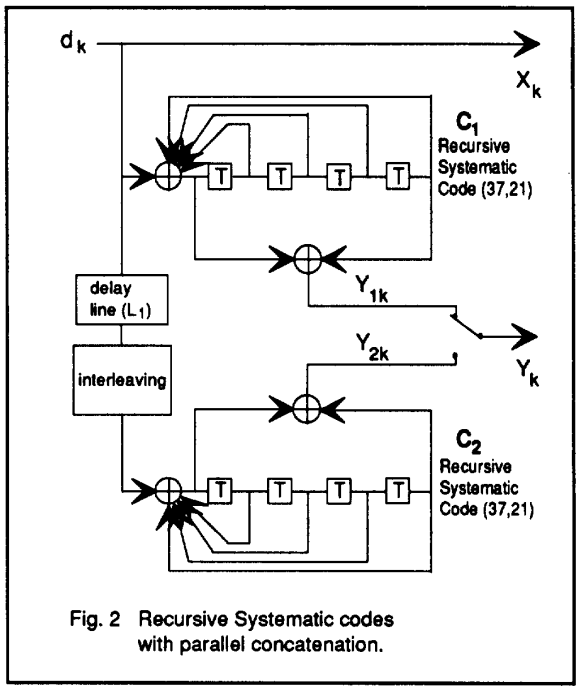


Fig. 2 Recursive Systematic codes with parallel concatenation.

The decoder **DEC** depicted in Fig. 3a, is made up of two elementary decoders ($DEC_1$ and $DEC_2$) in a serial concatenation scheme. The first elementary decoder $DEC_1$ is associated with the lower rate $R_1$ encoder $C_1$ and yields a soft (weighted) decision. The error bursts at the decoder $DEC_1$ output are scattered by the interleaver and the encoder delay $L_1$ is inserted to take the decoder $DEC_1$ delay into account. Parallel concatenation is a very attractive scheme because both elementary encoder and decoder use a single frequency clock.

For a discrete memoryless gaussian channel and a binary modulation, the decoder **DEC** input is made up of a couple $R_k$ of two random variables $x_k$ and $y_k$, at time $k$

$$x_k = (2d_k - 1) + i_k \qquad (7a)$$

$$y_k = (2Y_k - 1) + q_k, \qquad (7b)$$

where $i_k$ and $q_k$ are two independent noises with the same variance $\sigma^2$. The redundant information $y_k$ is demultiplexed and sent to decoder $DEC_1$ when $Y_k = Y_{1k}$ and toward decoder $DEC_2$ when $Y_k = Y_{2k}$. When the redundant information of a given encoder ($C_1$ or $C_2$) is not emitted, the corresponding decoder input is set to zero. This is performed by the DEMUX/INSERTION block.

It is well known that soft decoding is better than hard decoding, therefore the first decoder $DEC_1$ must deliver to the second decoder $DEC_2$ a weighted (soft) decision. The Logarithm of Likelihood Ratio (LLR), $\Lambda_1(d_k)$ associated with each decoded bit $d_k$ by the first decoder $DEC_1$ is a relevant piece of information for the second decoder $DEC_2$

$$\Lambda_1(d_k) = Log \frac{P_r\{d_k = 1/observation\}}{P_r\{d_k = 0/observation\}}. \qquad (8)$$

where $P_r\{d_k = i\ /observation\}$, $i = 0, 1$ is the *a posteriori probability* (APP) of the data bit $d_k$.

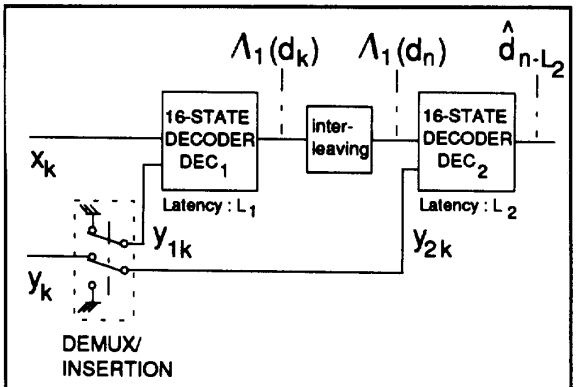


Fig. 3a Principle of the decoder according to a serial concatenation scheme.

## III - OPTIMAL DECODING OF RSC CODES WITH WEIGHTED DECISION

The VITERBI algorithm is an optimal decoding method which minimizes the probability of sequence error for convolutional codes. Unfortunately this algorithm is not able to yield the APP for each decoded bit. A relevant algorithm for this purpose has been proposed by BAHL *et al.* [1]. This algorithm minimizes the bit error probability in decoding linear block and convolutional codes and yields the APP for each decoded bit. For RSC codes, the BAHL *et al.* algorithm must be modified in order to take into account their recursive character.

### III - 1 Modified BAHL *et al.* algorithm for RSC codes

Consider a RSC code with constraint length $K$; at time $k$ the encoder state $S_k$ is represented by a $K$-uple

$$S_k = (a_k, a_{k-1} \ldots\ldots a_{k-K+1}). \qquad (9)$$

Also suppose that the information bit sequence $\{d_k\}$ is made up of $N$ independent bits $d_k$, taking values 0 and 1 with equal probability and that the encoder initial state $S_0$ and final state $S_N$ are both equal to zero, *i.e*

$$S_0 = S_N = (0, 0 \ldots\ldots 0) = \mathbf{0}. \qquad (10)$$

The encoder output codeword sequence, noted $C_1^N = \{C_1 \ldots\ldots C_k \ldots\ldots C_N\}$ is the input to a discrete gaussian memoryless channel whose output is the sequence $R_1^N = \{R_1 \ldots\ldots R_k \ldots\ldots R_N\}$ where $R_k = (x_k, y_k)$ is defined by relations (7a) and (7b).

The APP of a decoded data bit $d_k$ can be derived from the joint probability $\lambda_k^i(m)$ defined by

$$\lambda_k^i(m) = P_r\left\{d_k = i, S_k = m / R_1^N\right\} \qquad (11)$$

and thus, the APP of a decoded data bit $d_k$ is equal to

$$P_r\left\{d_k = i / R_1^N\right\} = \sum_m \lambda_k^i(m), \; i = 0,1. \quad (12)$$

From relations (8) and (12), the LLR $\Lambda(d_k)$ associated with a decoded bit $d_k$ can be written as

$$\Lambda(d_k) = Log \frac{\sum_m \lambda_k^1(m)}{\sum_m \lambda_k^0(m)}. \qquad (13)$$

Finally the decoder can make a decision by comparing $\Lambda(d_k)$ to a threshold equal to zero

$$\hat{d}_k = 1 \quad if \quad \Lambda(d_k) > 0$$

$$\hat{d}_k = 0 \quad if \quad \Lambda(d_k) < 0. \qquad (14)$$

In order to compute the probability $\lambda_k^i(m)$, let us introduce the probability functions $\alpha_k^i(m)$, $\beta_k(m)$ and $\gamma_i(R_k, m', m)$

$$\alpha_k^i(m) = \frac{P_r\left\{d_k = i, S_k = m, R_1^k\right\}}{P_r\left\{R_1^k\right\}} P_r\left\{d_k = i, S_k = m / R_1^k\right\} \quad (15)$$

$$\beta_k(m) = \frac{P_r\left\{R_{k+1}^N / S_k = m\right\}}{P_r\left\{R_{k+1}^N / R_1^k\right\}} \qquad (16)$$

$$\gamma_i(R_k, m', m) = P_r\left\{d_k = i, R_k, S_k = m / S_{k-1} = m')\right\}. \quad (17)$$

The joint probability $\lambda_k^i(m)$ can be rewritten using BAYES rule

$$\lambda_k^i(m) = \frac{P_r\left\{d_k = i, S_k = m, R_1^k, R_{k+1}^N\right\}}{P_r\left\{R_1^k, R_{k+1}^N\right\}}. \quad (18)$$

Thus we obtain

$$\lambda_k^i(m) = \frac{P_r\left\{d_k = i, S_k = m, R_1^k\right\}}{P_r\left\{R_1^k\right\}} \frac{P_r\left\{R_{k+1}^N / d_k = i, S_k = m, R_1^k\right\}}{P_r\left\{R_{k+1}^N / R_1^k\right\}}. \qquad (19)$$

Taking into account that events after time $k$ are not influenced by observation $R_1^k$ and bit $d_k$ if state $S_k$ is known, the probability $\lambda_k^i(m)$ is equal

$$\lambda_k^i(m) = \alpha_k^i(m)\beta_k(m). \qquad (20)$$

The probabilities $\alpha_k^i(m)$ and $\beta_k(m)$ can be recursively calculated from probability $\gamma_i(R_k, m', m)$. From annex I, we obtain

$$\alpha_k^i(m) = \frac{\sum_{m'} \sum_{j=0}^{1} \gamma_i(R_k, m', m)\alpha_{k-1}^j(m')}{\sum_m \sum_{m'} \sum_{i=0}^{1} \sum_{j=0}^{1} \gamma_i(R_k, m', m)\alpha_{k-1}^j(m')} \qquad (21)$$

and

$$\beta_k(m) = \frac{\sum_{m'} \sum_{i=0}^{1} \gamma_i(R_{k+1}, m, m')\beta_{k+1}(m')}{\sum_m \sum_{m'} \sum_{i=0}^{1} \sum_{j=0}^{1} \gamma_i(R_{k+1}, m', m)\alpha_k^j(m')}. \qquad (22)$$

The probability $\gamma_i(R_k, m', m)$ can be determined from transition probabilities of the discrete gaussian memoryless channel and transition probabilities of the encoder trellis. From relation (17), $\gamma_i(R_k, m', m)$ is given by

$$\gamma_i(R_k, m', m) = p(R_k / d_k = i, S_k = m, S_{k-1} = m')$$
$$q(d_k = i / S_k = m, S_{k-1} = m')\pi(S_k = m / S_{k-1} = m') \quad (23)$$

where p(./.) is the transition probability of the discrete gaussian memoryless channel. Conditionally to $(d_k = i, S_k = m, S_{k-1} = m')$, $x_k$ and $y_k$ are two uncorrelated gaussian variables and thus we obtain

$$p(R_k / d_k = i, S_k = m, S_{k-1} = m') =$$
$$p(x_k / d_k = i, S_k = m, S_{k-1} = m')$$
$$p(y_k / d_k = i, S_k = m, S_{k-1} = m'). \quad (24)$$

Since the convolutional encoder is a deterministic machine, $q(d_k = i / S_k = m, S_{k-1} = m')$ is equal to 0 or 1. The transition state probabilities $\pi(S_k = m / S_{k-1} = m')$ of the trellis are defined by the encoder input statistic. Generally, $P_r\left\{d_k = 1\right\} = P_r\left\{d_k = 0\right\} = 1/2$ and since there are two possible transitions from each state, $\pi(S_k = m / S_{k-1} = m') = 1/2$ for each of these transitions.

**Different steps of modified BAHL *et al.* algorithm**

-*Step* 0 : Probabilities $\alpha_0^i(m)$ and $\beta_N(m)$ are initialized according to relation (12)

$$\alpha_0^i(0) = 1 \;\; \alpha_0^i(m) = 0 \;\; \forall m \neq 0, \; i = 0,1 \qquad (25a)$$

$$\beta_N(0) = 1 \;\; \beta_N(m) = 0 \;\; \forall m \neq 0. \qquad (25b)$$

-*Step* 1 : For each observation $R_k$, the probabilities $\alpha_k^i(m)$ and $\gamma_i(R_k, m', m)$ are computed using relations (21) and (23) respectively.

-*Step* 2 : When the sequence $R_1^N$ has been completely received, probabilities $\beta_k(m)$ are computed using relation (22), and probabilities $\alpha_k^i(m)$ and $\beta_k(m)$ are multiplied in order to obtain $\lambda_k^i(m)$. Finally the LLR associated with each decoded bit $d_k$ is computed from relation (13).

## IV- THE EXTRINSIC INFORMATION OF THE RSC DECODER

In this chapter, we will show that the LLR $\Lambda(d_k)$ associated with each decoded bit $d_k$, is the sum of the LLR of $d_k$ at the decoder input and of another information called extrinsic information, generated by the decoder.

Using the LLR $\Lambda(d_k)$ definition (13) and relations (20) and (21), we obtain

$$\Lambda(d_k) = Log \frac{\sum_m \sum_{m'} \sum_{j=0}^{1} \gamma_1(R_k, m', m)\alpha_{k-1}^j(m')\beta_k(m)}{\sum_m \sum_{m'} \sum_{j=0}^{1} \gamma_0(R_k, m', m)\alpha_{k-1}^j(m')\beta_k(m)}. \quad (26)$$

Since the encoder is systematic ($X_k = d_k$), the transition probability $p(x_k / d_k = i, S_k = m, S_{k-1} = m')$ in expression $\gamma_i(R_k, m', m)$ is independent of state values $S_k$ and $S_{k-1}$. Therefore we can factorize this transition probability in the numerator and in the denominator of relation (26)

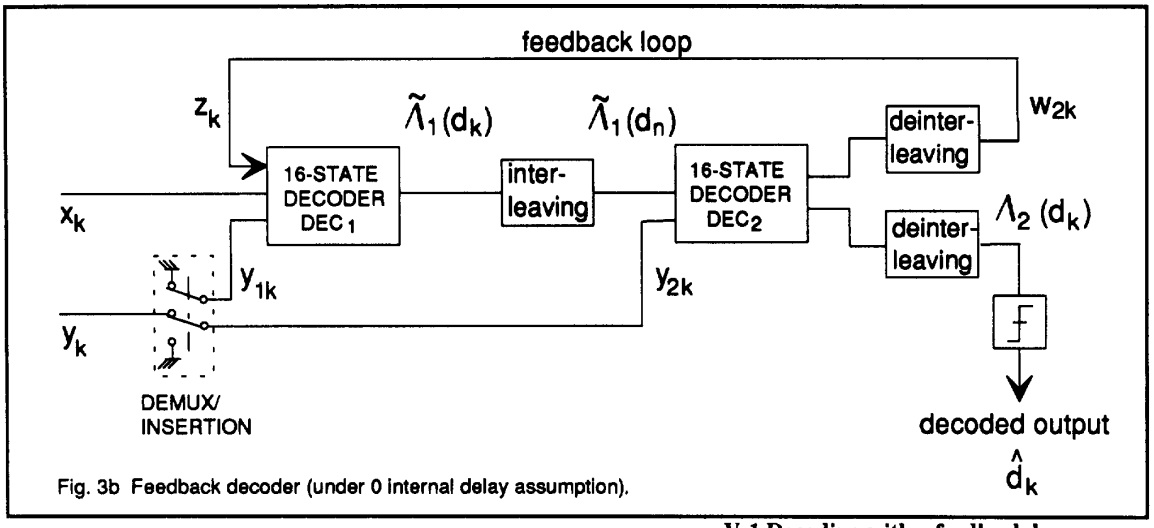


Fig. 3b Feedback decoder (under 0 internal delay assumption).

$$\Lambda(d_k) = Log\frac{p(x_k/d_k=1)}{p(x_k/d_k=0)} + Log\frac{\sum_m\sum_{m'}\sum_{j=0}^{1}\gamma_1(y_k,m',m)\alpha_{k-1}^{j}(m')\beta_k(m)}{\sum_m\sum_{m'}\sum_{j=0}^{1}\gamma_0(y_k,m',m)\alpha_{k-1}^{j}(m')\beta_k(m)}. \quad (27)$$

Conditionally to $d_k=1$ (resp. $d_k=0$), variables $x_k$ are gaussian with mean 1 (resp. -1) and variance $\sigma^2$, thus the LLR $\Lambda(d_k)$ is still equal to

$$\Lambda(d_k) = \frac{2}{\sigma^2}x_k + W_k \quad (28)$$

where

$$W_k = \Lambda(d_k)\big|_{x_k=0} = Log\frac{\sum_m\sum_{m'}\sum_{j=0}^{1}\gamma_1(y_k,m',m)\alpha_{k-1}^{j}(m')\beta_k(m)}{\sum_m\sum_{m'}\sum_{j=0}^{1}\gamma_0(y_k,m',m)\alpha_{k-1}^{j}(m')\beta_k(m)}. \quad (29)$$

$W_k$ is a function of the redundant information introduced by the encoder. In general $W_k$ has the same sign as $d_k$; therefore $W_k$ may improve the LLR associated with each decoded data bit $d_k$. This quantity represents the extrinsic information supplied by the decoder and does not depend on decoder input $x_k$. This property will be used for decoding the two parallel concatenated encoders.

## V - DECODING SCHEME OF PARALLEL CONCATENATION CODES

In the decoding scheme represented in Fig. 3a, decoder **DEC1** computes LLR $\Lambda_1(d_k)$ for each transmitted bit $d_k$ from sequences $\{x_k\}$ and $\{y_k\}$, then the decoder **DEC2** performs the decoding of sequence $\{d_k\}$ from sequences $\{\Lambda_1(d_k)\}$ and $\{y_k\}$. Decoder **DEC1** uses the modified BAHL *et al.* algorithm and decoder **DEC2** may use the VITERBI algorithm. The global decoding rule is not optimal because the first decoder uses only a fraction of the available redundant information. Therefore it is possible to improve the performance of this serial decoder by using a feedback loop.

### V-1 Decoding with a feedback loop

We consider now that both decoders **DEC1** and **DEC2** use the modified BAHL *et al.* algorithm. We have seen in section IV that the LLR at the decoder output can be expressed as a sum of two terms if the decoder inputs were independent. Hence if the decoder **DEC2** inputs $\Lambda_1(d_k)$ and $y_{2k}$ are independent, the LLR $\Lambda_2(d_k)$ at the decoder **DEC2** output can be written as

$$\Lambda_2(d_k) = f(\Lambda_1(d_k)) + W_{2k} \quad (30)$$

with

$$\Lambda_1(d_k) = \frac{2}{\sigma^2}x_k + W_{1k} \quad (31)$$

From relation (29), we can see that the decoder **DEC2** extrinsic information $W_{2k}$ is a function of the sequence $\{\Lambda_1(d_n)\}_{n\neq k}$. Since $\Lambda_1(d_n)$ depends on observation $R_1^N$, extrinsic information $W_{2k}$ is correlated with observations $x_k$ and $y_{1k}$. Nevertheless from relation (29), the greater $|n-k|$ is, the less correlated are $\Lambda_1(d_n)$ and observations $x_k$, $y_k$. Thus, due to the presence of interleaving between decoders **DEC1** and **DEC2**, extrinsic information $W_{2k}$ and observations $x_k$, $y_{1k}$ are weakly correlated. Therefore extrinsic information $W_{2k}$ and observations $x_k$, $y_{1k}$ can be jointly used for carrying out a new decoding of bit $d_k$, the extrinsic information $z_k = W_{2k}$ acting as a diversity effect in an iterative process.

In Fig. 3b, we have depicted a new decoding scheme using the extrinsic information $W_{2k}$ generated by decoder **DEC2** in a feedback loop. This decoder does not take into account the different delays introduced by decoder **DEC1** and **DEC2** and a more realistic decoding structure will be presented later.

The first decoder **DEC1** now has three data inputs, ($x_k$, $y_{1k}$, $z_k$) and probabilities $\alpha_{1k}^{i}(m)$ and $\beta_{1k}(m)$ are computed in substituting $R_k=\{x_k, y_{1k}\}$ by $R_k=(x_k, y_{1k}, z_k)$ in relations (21) and (22). Taking into account that $z_k$ is weakly correlated with $x_k$ and $y_{1k}$ and supposing that $z_k$ can be approximated by a gaussian variable with variance $\sigma_z^2 \neq \sigma^2$, the transition probability of the discrete gaussian memoryless channel can be now factored in three terms

$p(R_k/d_k = i, S_k = m, S_{k-1} = m') = p(x_k/.)p(y_k/.)p(z_k/.)$ (32)

The encoder $\mathbf{C_1}$ with initial rate $R_1$, through the feedback loop, is now equivalent to a rate $R'_1$ encoder with

$$R'_1 = \frac{R_1}{1 + R_1}. \qquad (33)$$

The first decoder obtains an additional redundant information with $z_k$ that may significantly improve its performances; the term *Turbo-codes* is given for this iterative decoder scheme with reference to the turbo engine principle.

With the feedback decoder, the LLR $\Lambda_1(d_k)$ generated by decoder $\mathbf{DEC_1}$ is now equal to

$$\Lambda_1(d_k) = \frac{2}{\sigma^2} x_k + \frac{2}{\sigma_z^2} z_k + W_{1k} \qquad (34)$$

where $W_{1k}$ depends on sequence $\{z_n\}_{n \neq k}$. As indicated above, information $z_k$ has been built by decoder $\mathbf{DEC_2}$ at the previous decoding step. Therefore $z_k$ must not be used as input information for decoder $\mathbf{DEC_2}$. Thus decoder $\mathbf{DEC_2}$ input sequences at step $p$ $(p \geq 2)$ will be sequences $\left\{\tilde{\Lambda}_1(d_n)\right\}$ and $\{y_{2k}\}$ with

$$\tilde{\Lambda}_1(d_n) = \Lambda_1(d_n)_{z_n = 0}. \qquad (35)$$

Finally from relation (30), decoder $\mathbf{DEC_2}$ extrinsic information $z_k = W_{2k}$, after deinterleaving, can be written as

$$z_k = W_{2k} = \Lambda_2(d_k)\big|_{\tilde{\Lambda}_1(d_k)=0} \qquad (36)$$

and the decision at the decoder DEC output is

$$\hat{d}_k = sign\left[\Lambda_2(d_k)\right]. \qquad (37)$$

The decoding delays introduced by decoder $\mathbf{DEC}$ ($\mathbf{DEC = DEC_1 + DEC_2}$), the interleaver and the deinterleaver imply that the feedback information $z_k$ must be used through an iterative process as represented in Fig. 4a, 4b. In fact, the global decoder circuit is composed of $P$ pipelined identical elementary decoders (Fig. 4a). The $p$th decoder $\mathbf{DEC}$ (Fig. 4b) input, is made up of demodulator output sequences $(x)_p$ and $(y)_p$ through a delay line and of extrinsic information $(z)_p$ generated by the $(p\text{-}1)$th decoder $\mathbf{DEC}$. Note that the variance $\sigma_z^2$ of the extrinsic information and the variance of $\tilde{\Lambda}_1(d_k)$ must be estimated at each decoding step $p$.

### V-2 Interleaving

The interleaver uses a square matrix and bits $\{d_k\}$ are written row by row and read pseudo-randomly. This non-uniform reading rule is able to spread the residual error blocks of rectangular form, that may set up in the interleaver located behind the first decoder $\mathbf{DEC_1}$, and to give the greater free distance as possible to the concatenated (parallel) code.

## VI - RESULTS

For a rate $R$=1/2 encoder with constraint length $K$=5, generators $G_1$=37, $G_2$=21 and parallel concatenation ($R_1$=$R_2$=2/3), we have computed the Bit Error Rate (BER) after each decoding step using the Monte Carlo method, as a function of signal to noise ratio $E_b/N_0$ where $E_b$ is the energy received per information bit $d_k$ and $N_0$ is the noise monolateral power spectral density. The interleaver consists of a 256x256 matrix and the modified BAHL *et al.* algorithm has been used with length data block of $N$=65536 bits. In

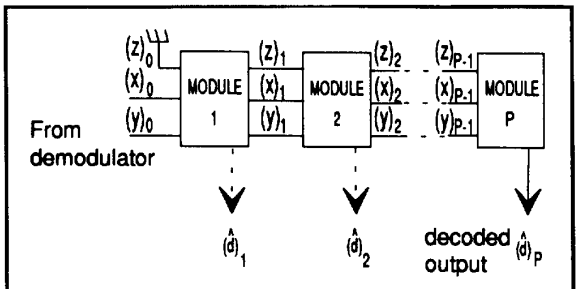


Fig. 4a Modular pipelined decoder, corresponding to an iterative processus of the feedback decoding.

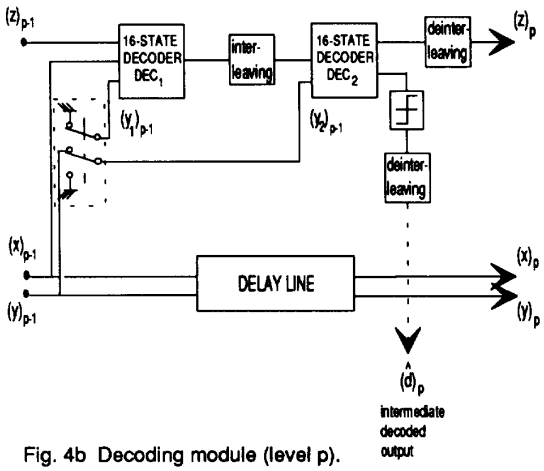


Fig. 4b Decoding module (level p).

order to evaluate a BER equal to $10^{-5}$, we have considered 128 data blocks i.e. approximatively 8 x$10^6$ bits $d_k$. The BER versus $E_b/N_0$, for different values of $p$ is plotted in Fig. 5. For any given signal to noise ratio greater than 0 dB, the BER decreases as a function of the decoding step $p$. The coding gain is fairly high for the first values of $p$ ($p$=1,2,3) and carries on increasing for the subsequent values of $p$. For $p$=18 for instance, the BER is lower than $10^{-5}$ at $E_b/N_0$= 0,7 dB. Remember that the Shannon limit for a binary modulation with $R$=1/2, is $P_e$ = 0 (several authors take $P_e$ =$10^{-5}$ as a reference) for $E_b/N_0$= 0 dB. With parallel concatenation of RSC convolutional codes and feedback decoding, the performances are at 0,7 dB from Shannon's limit.

The influence of the constraint length on the BER has also been examined. For $K$ greater than 5, at $E_b/N_0$= 0,7 dB, the BER is slightly worst at the first ($p$=1) decoding step and the feedback decoding is inefficient to improve the final BER. For $K$ smaller than 5, at $E_b/N_0$= 0,7 dB, the BER is slightly better at the first decoding step than for $K$ equal to 5, but the correction capacity of encoders $\mathbf{C_1}$ and $\mathbf{C_2}$ is too weak to improve the BER with feedback decoding. For $K$=4 (*i.e.* 8-state elementary decoders) and after iteration 18, a BER of $10^{-5}$ is achieved at $E_b/N_0$ = 0,9 dB. For $K$ equal to 5, we have tested several generators ($G_1$, $G_2$ ) and the best results were achieved with $G_1$=37, $G_2$=21.

DOCKET ALARM

# Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

## Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

## Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

## Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

## API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

**LAW FIRMS**
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

**FINANCIAL INSTITUTIONS**
Litigation and bankruptcy checks for companies and debtors.

**E-DISCOVERY AND LEGAL VENDORS**
Sync your system to PACER to automate legal marketing.