ELEC 6131: Error Detecting and Correcting Codes

Instructor:
Dr. M. R. Soleymani, Office: EV-5.125, Telephone: 848-2424 ext: 4103.
Time and Place: Tuesday, 17:45 – 20:15.
Office Hours: Tuesday, 15:00 – 17:00

## LECTURE 10: Concatenated Coding and Turbo Codes

Concatenated Coding

▶ As the error correcting capability of codes increases, i.e., as their rate is reduced their decoding becomes more complex. For example, if we use a (255,239) RS code, we have to solve a polynomial of degree 8 to find the 8 error byte's location. But if we use a (255,205) code, the number of errors that can be corrected is increased to 25. So, a polynomial of degree 25 has to be found and solved.

Concatenated Coding

▶ Apart from rate, the length of code has usually an exponential effect on the complexity of decoding. And it is evident that in order to get good coding gain, we need to have long codewords.

▶ In order to avoid the complexity of decoding, Forney suggested the use of concatenated codes. In this scheme the data is coded with one encoder (called the outer encoder) and then with another (inner encoder). The decoding is performed in the reverse order

Concatenated Coding

▶ The following figure shows the block diagram of a concatenated coding scheme:
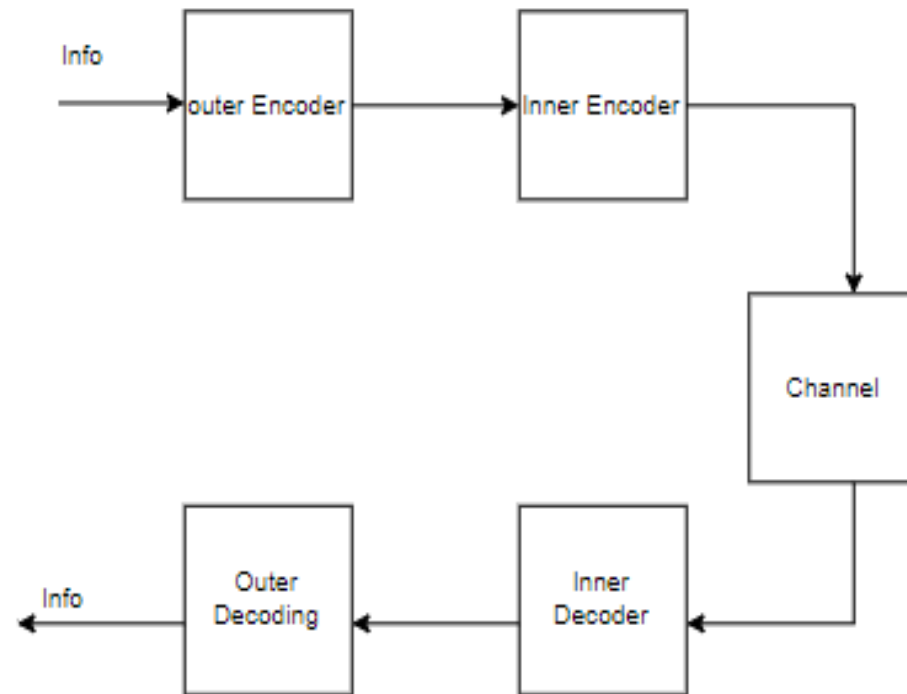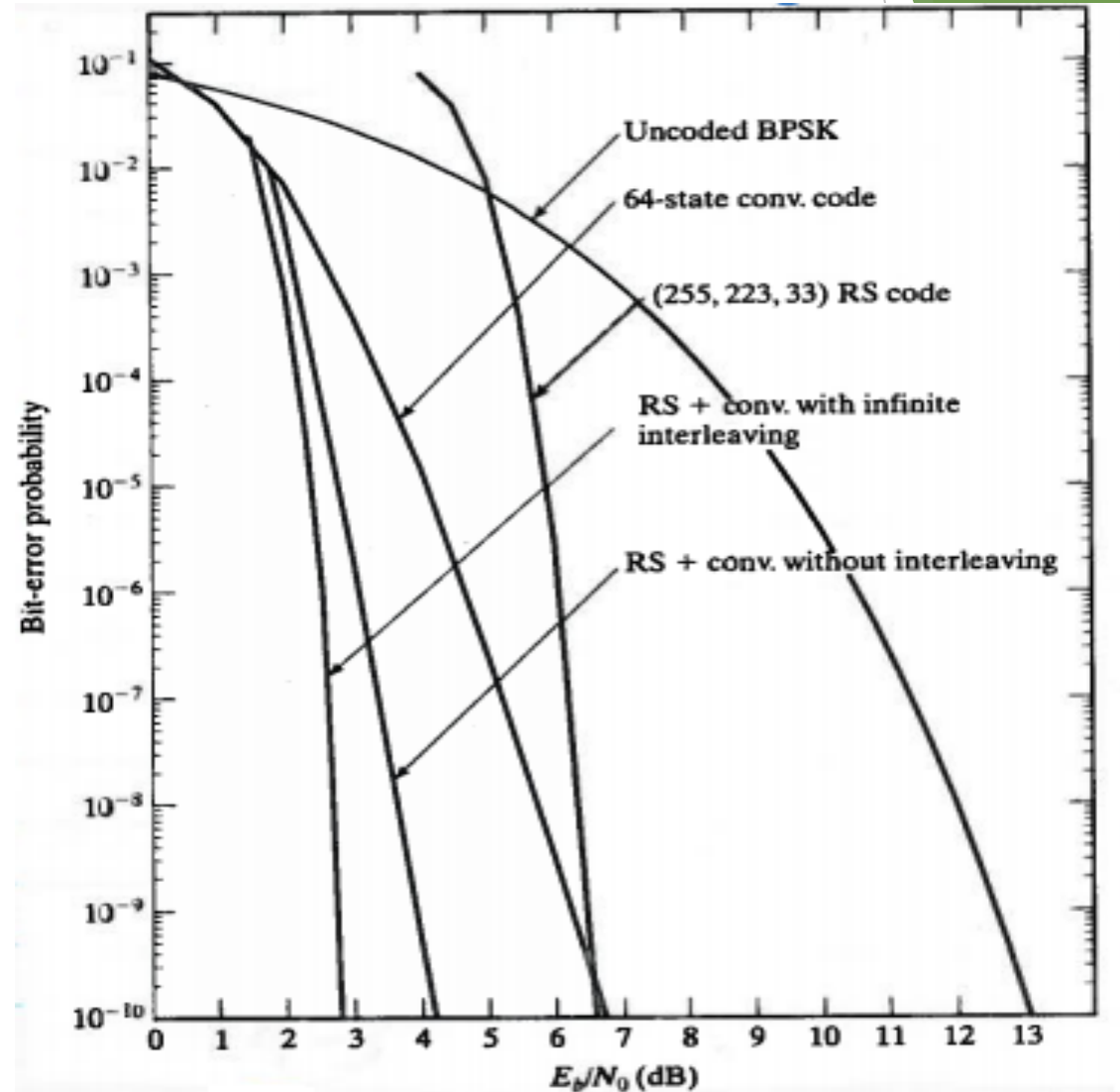
Concatenated Coding

▶ It is possible to use more than two stages of coding. The inner and outer codes can be either both block codes or one block code and another convolutional code, or some other arrangement.

▶ As we saw in the previous lecture, convolutional codes create bursts of errors. Concatenating them (as the inner code) with some burst error correcting block code such as RS code is a good choice.

Concatenated Coding

▶ Type equation here.One combination frequently used by industry is to use convolutional codes as the inner code and Reed-Solomon codes as the outer code. One example is NASA's Tracking Data Relay Satellite System (TDRSS) that uses a (255,223) RS code as the outer code and a 64-state convolutional code with $g_1(D) = 1 + D + D^3 + D^4 + D^6$ and $g_2(D) = 1 + D^3 + D^4 + D^5 + D^6$. This is the industry standard (133,171) code. The overall rate is $\frac{223}{255} \times \frac{1}{2} = 0.437$.

Concatenated Coding

► Performance of this scheme
(with and without interleaving)
is compared with uncoded case
as well as just convolutional
or just RS coding.

Turbo Codes

▶ One possibility for improving the performance of concatenated codes is through <u>iterative</u> cooperation of inner and outer code to improve the decoding accuracy. In such situation, instead of decoding the inner code and giving the <u>hard</u> decision to the outer code, one can extract <u>soft values</u> from the inner code, i.e. , likelihood ratios and pass them to the outer decoder. Then outer decoder instead of starting from no prior information, that is, assuming $P(0) = P(1) = \frac{1}{2}$ can start from somewhere closer to actual situation. The outer decoder can then use this information to extract new likelihood ratios and passing them to the inner decoder. This process can be iterated until a steady-state is reached. While this is possible with a serial concatenation and any combination of codes, the best result can be achieved by parallel concatenation of two convolutional codes.
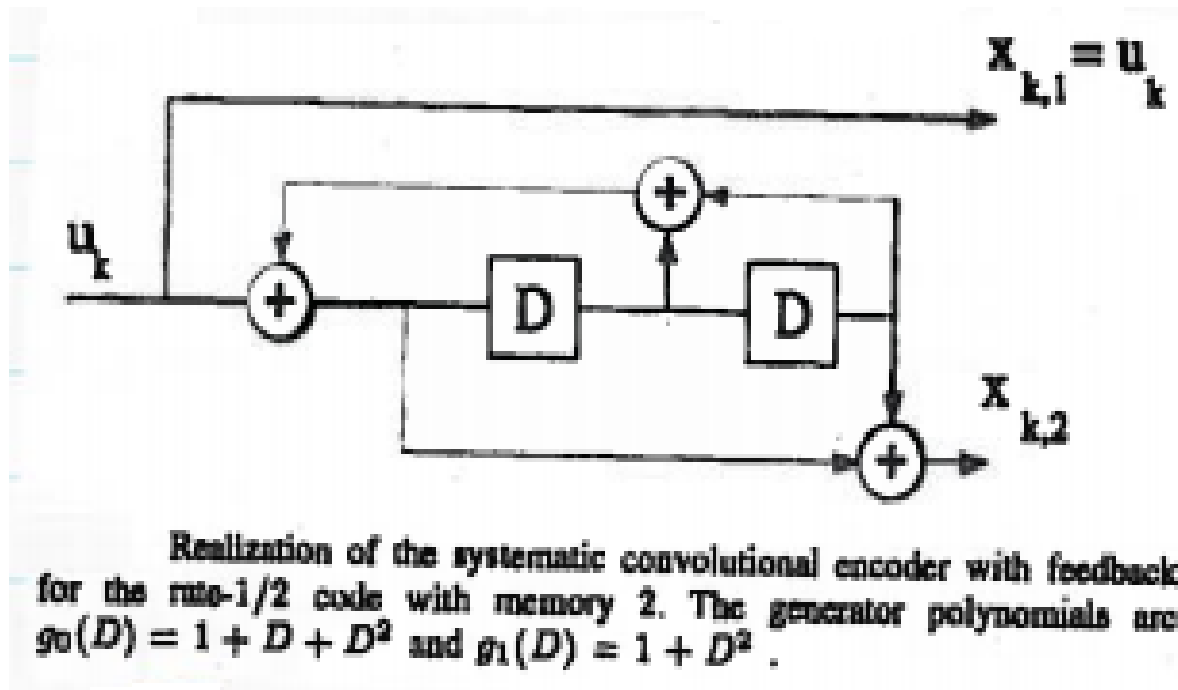
Turbo Codes

▶ In 1993 Berrow, Glavieux and Thitimajshima showed that using two very simple convolutional codes working on two copies of the input data (one being a permutation of the other), one can get results very close to Shannon's bounds.

▶ The encoder has the following structure
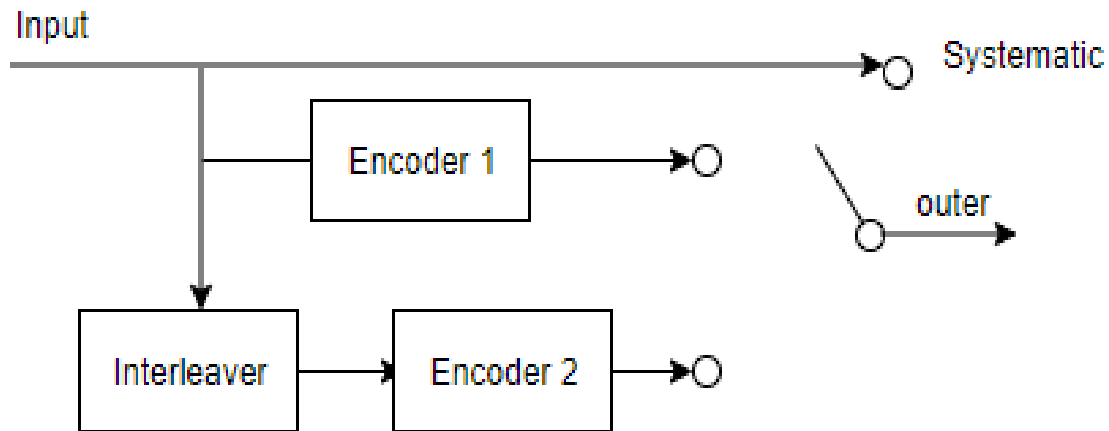
Turbo Codes

- The paper by Berrou et al. used a two memory (4-state) Recursive Systematic Convolutional code.

Realization of the systematic convolutional encoder with feedback for the rate-1/2 code with memory 2. The generator polynomials are $g_0(D) = 1 + D + D^2$ and $g_1(D) = 1 + D^2$.

Turbo Codes

▶ The above code sends the systematic part, only from one encoder, plus two parity streams. So, the rate of the code is $\frac{1}{3}$. However, by puncturing the parity streams, one can get higher rates. For example, if we send one systematic bit and a parity frame one of the two on each output, i.e.,

$$u, p_1, u, p_2, u, p_1, u, p_2, \ldots$$

▶ We have a rate $\frac{1}{2}$ code. While the constituent encoders each have only four states, the interleaver causes the output stream to have large minimum free distance.

Turbo Codes

▶ To <u>decode</u> the turbo code, we need two SISO (Soft-Input-Soft-Output) decoders.
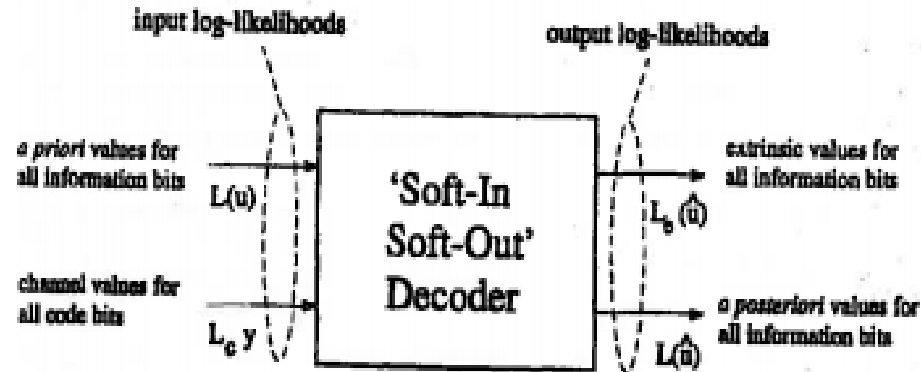
▶ Each SISO decoder works on one of the two streams. It takes as input the channel value, y, and the probability of the corresponding systematic bit being one (or equivalently, likelihood ratio) and generates its own likelihood ratio.
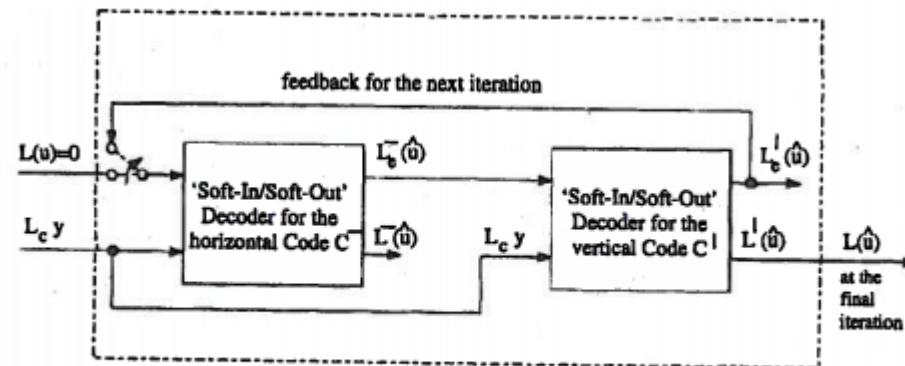
Iterative decoding procedure with two "soft-in/soft-out" decoders with initial $L(u) = 0$, i.e., equally likely source (information) bits.

Turbo Codes

▶ It then passes it's result i.e., likelihood ratio (or log-likelihood ratio) or the improvement if has added $L_e(\hat{u})$ called extrinsic log-likelihood ratio to the other decoder.
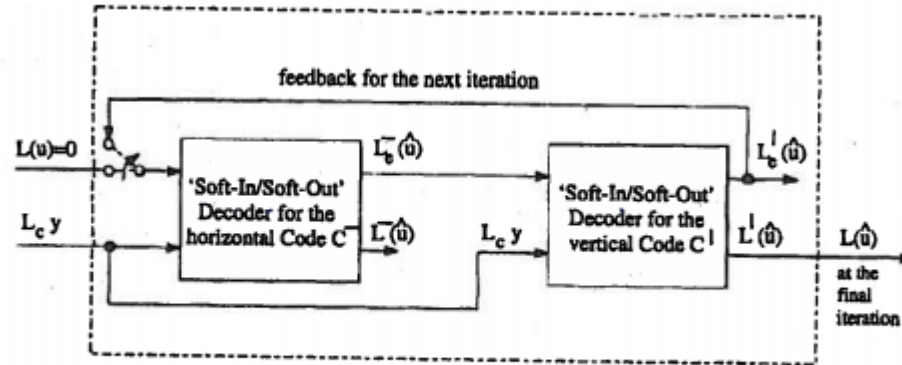
Iterative decoding procedure with two "soft-in/soft-out" decoders with initial $L(u) = 0$, i.e., equally likely source (information) bits.

▶ At the beginning L(u)=0, i.e., we assume that $p(0) = p(1) = \frac{1}{2}$ so $\log \frac{p(u=0)}{p(u=1)} = \log \frac{\frac{1}{2}}{\frac{1}{2}} = 0$. But in other iterations, $L^-(\hat{u})$ and $L'(\hat{u})$ are evolved, iteratively.

Decoding of Turbo Codes

▶ The <u>Soft-Value</u> of a bit is given as its log-likelihood ratio:

$$L(u) = log \frac{p(u = 0)}{p(u = 1)}$$

▶ Assume that we encode a sequence of k bits using an (n, k) systematic code. k of the encoded bits of the code sequence say x are equal to the values of u. the other $n - k$ are parity bits.

▶ Now assume that we assign ± 1 to bits of x,

i.e., x=0 →+1 and x=1→ -1.

▶ Assume that we transmit x's over an Additive White Gaussian Noise (AWGN) channel with possibly multiplying the symbols by a fade factor a.

▶ The conditional soft-value or log-likelihood ratio of x given y (the matched filter output) is:

$$L(x|y) = log \frac{p(x = +1|y)}{p(x = -1|y)} = log \left[\frac{p(x = +1)p(y|x = +1)}{p(x = -1)p(y|x = -1)} \right]$$

Decoding of Turbo Codes

▶ For a Gaussian Channel with fade factor a:

$$p(y|x = +1) = \frac{1}{\sqrt{\pi \frac{N_0}{E_s}}} e^{-\frac{E_s}{N_0}(y-\alpha)^2}$$

and

$$p(y|x = -1) = \frac{1}{\sqrt{\pi \frac{N_0}{E_s}}} e^{-\frac{E_s}{N_0}(y+\alpha)^2}$$

So:

$$L(x|y) = log \frac{e^{-\frac{E_s}{N_0}(y-\alpha)^2}}{e^{-\frac{E_s}{N_0}(y+\alpha)^2}} + log \frac{p(x = +1)}{p(x = -1)}$$

Or

$$L(x|y) = 4a \frac{E_s}{N_0} . y + L(x)$$
$$= L_c . y + L(x)$$

▶ For an AWGN channel $a = 1$.

Decoding of Turbo Codes

▶ A maximum a Posteriori (MAP) decoder outputs an a posteriori (i.e., after observing the total output of the matched filter $\underline{y} = (y_1, y_2, ...)$ log-likelihood ratio for transmitted +1 and -1:

$$L(\hat{u}) = L\left(u\middle|\underline{y}\right) = log \frac{p(u = +1|\underline{y})}{p(u = -1|\underline{y})}$$

▶ A MAP decoder uses a priori values L(u) for all information bits and the channel values $L_c . y$. It generates for each information bit a soft value $L(\hat{u})$ that in addition to $L_c . y$ and L(u) has $L_c(\hat{u})$ called the extrinsic information that depends on the structure of the encoder.

$$L(\hat{u}) = l_c . y + l(u) + l_e(\hat{u})$$

▶ The first decoder starts with $L_c . y$ only (at the beginning) and generate the extrinsic information

$$L_e^-(\hat{u}) = L^-(\hat{u}) - l_c . y$$

▶ The second decoder uses $l_e(\hat{u})$ and generates

$$L_e'(\hat{u}) = L'(\hat{u}) - (l_c . y + l_e^-(\hat{u})$$

Decoding of Turbo Codes

▶ This continues iteratively. At the end

$$L(\hat{u}) = l_c.y + l_e^-(\hat{u}) + l_e^-(\hat{u})$$

is the output and is used for making the final (hard) decision.

▶ At this point we discuss the generation of $L(\hat{u}_k)$ for all k for Convolutional codes.

▶ Consider the trellis diagram of a systematic Convolutional code given as:

Trellis structure of systematic convolutional codes with feedback encoders.
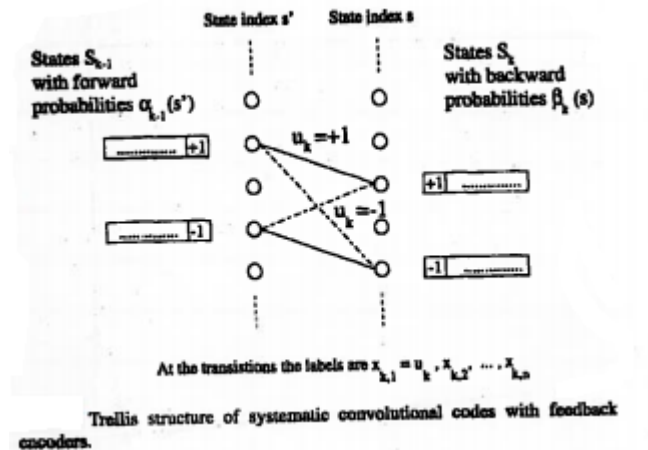
Decoding of Turbo Codes

▶ Then

$$L(\hat{u}_k) = log \frac{p(u_k = +1|\underline{y})}{p(u_k = -1|\underline{y})} = log \frac{\sum_{s,s',u_k=+1} p(s',s,\underline{y})}{\sum_{s,s',u_k=-1} p(s',s,y)}$$

Where states s' and s define bit u_k and the coded bit $x_{k,\gamma}$, for γ= 2,…,n.

▶ The sum of probabilities p$(s',s,\underline{y})$ are taken over all existing transitions from $s'\ to\ s$ with transition label $u_k = +1$ (in numerator) and with label $u_k = -1$ (in the denominator).

$$p\left(s',s,\underline{y}\right) = p\left(s',\underline{y}_{j<k}\right).p\left(s,\underline{y}_k\middle|s'\right).p(\underline{y}_{j>k}|s)$$
$$= p\left(s',\underline{y}_{j<k}\right).p(s|s').p(\underline{y}_k|s',s)p(\underline{y}_{j>k}|s)$$
$$= \alpha_{k-1}(s').\gamma_k(s',s).\beta_k(s)$$

Decoding of Turbo Codes

▶ $y_{j<k}$ is the sequence of received symbols from the beginning up to time k-1, $y_{j>k}$ is the sequence from k+1 to the end.

▶ α_k 's can be found using the forward recursion.

$$\alpha_k(s) = \sum_{s'} \gamma_k(s', s)\alpha_{k-1}(s')$$

▶ And β_k's are found using backward recursion

$$\beta_{k-1}(s') = \sum_{s} \gamma_k(s', s).\beta_k(s)$$

▶ The branch transition probabilities are given by,

$$\gamma_k(s', s) = p\left(\underline{y_k}\middle|u_k\right).p(u_k)$$

Decoding of Turbo Codes

$$L(\hat{u}_k) = L_c y_{k,1} + L(u_k) + log \frac{\sum_{\substack{(s',s) \\ uk=+1}} \gamma_k^{(e)}(s',s)\alpha_{k-1}^{(s')}\beta_k(s)}{\sum_{\substack{(s',s) \\ uk=-1}} \gamma_k^{(e)}(s',s)\alpha_{k-1}^{(s')}\beta_k(s)}$$

▶ **Log-max algorithm:**

▶ Since the summations in the above formula are difficult to perform, the following fact is used to reduce the complexity:

$$\log(x + y) = logx\left(1 + \frac{y}{x}\right) \simeq \log x \text{ if } \frac{y}{x} \ll 1 \text{ or } y \ll x. \text{ That is}$$
$$\log(x + y) = logmax(x, y)$$

▶ So, the approximate version of MAP or Max-log MAP is:
$$L(\hat{u})$$
$$= L_c.y_k + L(u_k) + \max_{\substack{s',s \\ u_k=+1}}\left[\log\alpha_{k-1}(s') + \log\beta_k(s) + log\gamma_k^{(e)}(s',s)\right]$$
$$- \max_{\substack{s',s \\ u_k=-1}}\left[log\alpha_{k-1}(s') + log\beta_k(s) + log\gamma_k^{(e)}(s',s)\right].$$

Turbo Codes: Performance Results

▶ Following shows the performance of rate $\frac{1}{2}$ Turbo code with 1,2,....,18 iterations. Each iteration involves two decodings, one by each decoder and exchange of information.
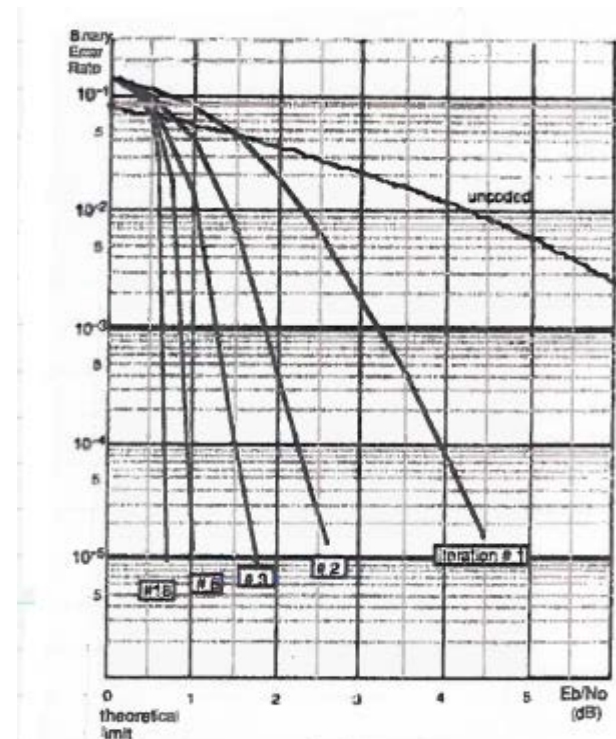

Fig. 9. BER given by iterative decoding ($p = 1, \cdots 18$) of a rate $R = 1/2$ encoder, memory $\nu = 4$, generators $G_1 = 37, G_2 = 21$, with interleaving 256×256.