Turbo codes are a groundbreaking class of forward error correction (FEC) codes that revolutionized digital communication by approaching the Shannon limit, the theoretical maximum efficiency for reliable data transmission over noisy channels. Introduced in 1993 by Berrou, Glavieux, and Thitimajshima, these codes combine parallel concatenated convolutional codes with iterative decoding to achieve exceptional error-correction performance.

Background

Turbo codes emerged as a solution to the challenge of balancing computational complexity and error-correction capability. Traditional codes required impractical processing power for near-optimal performance, but turbo codes achieved this through recursive systematic convolutional (RSC) encoders and innovative decoding techniques. Their development marked a paradigm shift in coding theory, enabling reliable communication in bandwidth-constrained environments like satellite links and cellular networks.

Core Structure

The encoder structure consists of:
- Parallel RSC encoders: Two identical recursive systematic convolutional codes generate parity bits from the same input sequence.
- Interleaver: Rearranges input bits pseudo-randomly before the second encoder, creating statistical independence between parity streams.
- Systematic output: Transmits original data bits alongside parity information, maintaining a code rate typically between 1/3 and 1/2.

This parallel concatenation with interleaving produces code words with large effective block lengths while keeping individual component codes manageable.

Iterative Decoding

The decoding process employs:
- Soft-input/soft-output (SISO) decoders: Each component decoder uses algorithms like BCJR to estimate bit probabilities.
- Extrinsic information exchange: Decoders share reliability metrics (log-likelihood ratios) in successive iterations.
- Stopping criteria: Convergence occurs when decoders agree on the most probable codeword, typically within 15-18 iterations.

This feedback mechanism resembles solving a crossword puzzle through collaborative clue-checking between two solvers.

Performance Characteristics

- Near-Shannon limit operation: Achieves BER of 10^{-5} at $E_b/N_0 \approx 0.7$ dB for rate 1/2 codes.
- Distance spectrum optimization: The interleaver design critically influences minimum Hamming distance.
- Spectral efficiency: Competes with LDPC codes while offering implementation advantages in certain scenarios.

Applications

Turbo codes have been adopted in:

- 3G/4G mobile standards (UMTS, LTE)
- Deep-space communications (CCSDS)
- Digital video broadcasting (DVB)
- Satellite communication systems

Their introduction spurred advances in iterative processing techniques, influencing modern coding theory and paving the way for new hybrid coding approaches. Despite competition from LDPC codes, turbo codes remain relevant in systems requiring robust error correction with moderate latency constraints.

Citations:
[1] [PDF] Turbo Codes - ECE Department, Haldia Institute of Technology https://www.ecehithaldia.in/teaching_material/Turbo%20Codes502845964.pdf
[2] Key Concepts of Turbo Codes to Know for Coding Theory - Fiveable https://library.fiveable.me/lists/key-concepts-of-turbo-codes
[3] [PDF] chapter 12 - turbo codes - WVU IT Help Center https://community.wvu.edu/~mcvalenti/documents/DOWLA-CH12.pdf
[4] Turbo code - Wikipedia https://en.wikipedia.org/wiki/Turbo_code
[5] What are turbo codes? - I'MTech https://imtech.imt.fr/en/2016/09/16/what-are-turbo-codes/
[6] Key Concepts of Turbo Codes to Know for Coding Theory - Fiveable https://fiveable.me/lists/key-concepts-of-turbo-codes
[7] [PDF] Fundamentals of Turbo Codes https://faculty.uml.edu/jweitzen/16.548/ClassNotes/art_sklar3_turbocodes.pdf
[8] Turbo code - Scholarpedia http://www.scholarpedia.org/article/Turbo_code
[9] Understanding turbo codes: A signal processing study - ScienceDirect https://www.sciencedirect.com/science/article/pii/S2949715923000616
[10] [PDF] Part 2.4 Turbo codes - HKU https://www.eee.hku.hk/~sdma/elec7073/Part2-4-Turbo%20codes.pdf
[11] [PDF] Introduction to Turbo Coding and Turbo Detection https://www.southampton.ac.uk/~sqc/ELEC6214/AWCNS-L16.pdf
[12] [PDF] Lecture 29 - Introduction to Turbo Codes http://elearn.psgcas.ac.in/nptel/courses/video/108102117/lec29.pdf
[13] Turbo Code - an overview | ScienceDirect Topics https://www.sciencedirect.com/topics/mathematics/turbo-code

Summary:

Application of interest:

AWGN channels- Additive White Gaussian Channel i.e. only noise or impairment would be by addition of gaussian noise.

Some terms:

soft decision/ continuous value- real number like 1.23, 0.01, etc.
Hard decision/ abstract value- 1, 0
Siso- soft input soft output

Likelihood Functions:

“A posteriori probability” - in easy terms- Bayes’ theorem.

$$P\ (d\ =\ i\,|\,x) =\ \frac{p\,(x\,|\,d\ =\ i\,)\ \ P(d\ =\ i)}{p\,(x)} \quad i\ =\ 1\ ,\ .\ .\ .\ ,\ M \tag{1}$$

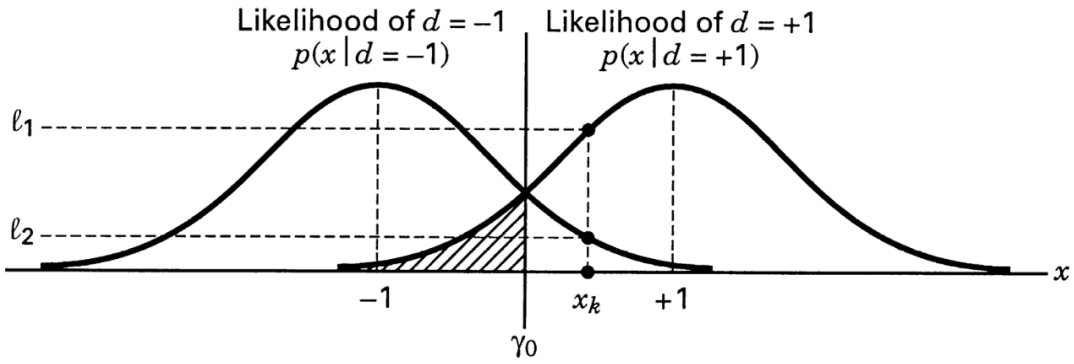
and

$$p(x) = \sum_{i=1}^{M}\ p\,(x\,|\,d\ =\ i)\ \ P\,(d\ =\ i) \tag{2}$$

Here, d=i represents data d which we sent over. It belongs to one of the signal classes(=1,-1,+5,-5 V or say 0,1 if binary) from a set of M classes(set of all signal classes).
p(x|d=i) represents probability density function of receiving signal x (x is data+ noise) given the input was d.
P(d=i) is called a priori probability of occurrence of ith signal class, basically P(d=heads)= ½ i.e. probability that heads was input? Like initial probability.

P(d=i|x) is the APP of continuous values random variable x.

Lets understand by two signal class case i.e. binary logical elements 1, 0 or +1, -1 Voltages respectively.
So the variable d is used to represent the transmitted data bit.



The x axis represents “x” generated at the receiver(input +noise) while the graph represents pdfs.

Suppose at kth time interval, x_k was observed. The “maximum likelihood decision rule” says that as l_1>l_2 at x_k, d_k=+1.

At gamma_not, it is not possible to use this rule for obvious reasons.
Using Bayes theorem, the decision rule is equivalent to talking about the probability ratios

$$P(d = +1|x) \underset{H_2}{\overset{H_1}{\gtrless}} P(d = -1|x)$$

$$p(x|d = +1)\ P(d = +1) \underset{H_2}{\overset{H_1}{\gtrless}} p(x|d = -1)\ P(d = -1)$$

$$\frac{p(x|d = +1)}{p(x|d = -1)} \underset{H_2}{\overset{H_1}{\gtrless}} \frac{P(d = -1)}{P(d = +1)} \quad \text{or} \quad \frac{p(x|d = +1)\ P(d = +1)}{p(x|d = -1)\ P(d = -1)} \underset{H_2}{\overset{H_1}{\gtrless}} 1$$

Where H1 and H2 are the two hypothesis as d=1 or -1.

Log Likelihood Ratios-

This motivates us to define the log- likelihood Ratios (LLR)s which will be of utmost importance here on.

$$L(d \mid x) = \log\left[\frac{P(d = +1 \mid x)}{P(d = -1 \mid x)}\right] = \log\left[\frac{p(x \mid d = +1)\ P(d = +1)}{p(x \mid d = -1)\ P(d = -1)}\right] \quad (6)$$

$$L(d \mid x) = \log\left[\frac{p(x \mid d = +1)}{p(x \mid d = -1)}\right] + \log\left[\frac{P(d = +1)}{P(d = -1)}\right] \quad (7)$$

$$L(d \mid x) = L(x \mid d) + L(d) \quad (8)$$

where $L(x|d)$ is the LLR of the test statistic x obtained by measurements of the channel output x under the alternate conditions that $d = +1$ or $d = -1$ may have been transmitted, and $L(d)$ is the a priori LLR of the data bit d.

To simplify the notation, Equation (8) is rewritten as follows:

$$L'(\hat{d}) = L_c(x) + L(d) \quad (9)$$

So L_c(x) is result of channel measurement made at receiver, L(d) is “a priori” of data bit d.
After receiver - L_c(x)
After demodulator-L’(d^)
After decoder- L_(d^)

$$L(\hat{d}) = L'(\hat{d}) + L_e(\hat{d})$$

L_e(d^) is called the extrinsic LLR and represents what we get extra out of decoding process (using parity bits?)

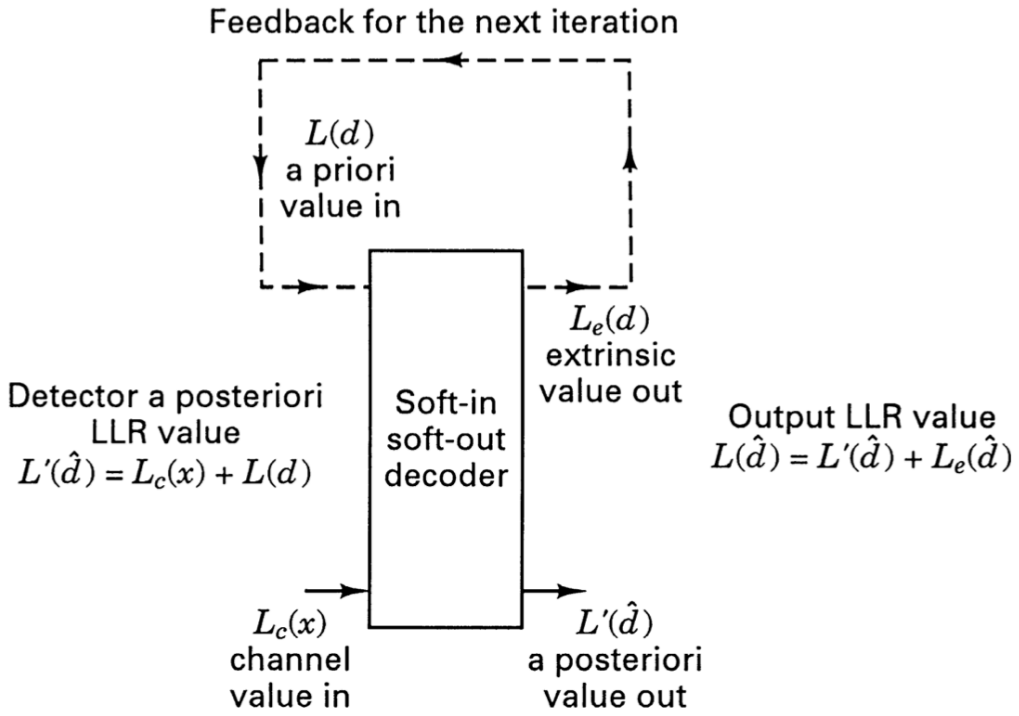
Thus the output LLR is written as

$$L(\hat{d}) = L_c(x) + L(d) + L_e(\hat{d})$$

It’s sign represents whether d= +1 or -1 and magnitude represents the confidence of this decision of decoder.

Basics of Iterative decoding:

Assuming binary data to be equally likely, i.e.e Initial L(d)=0,



This means in the first iteration initial input is L_c(x) and after the last iteration, last output is L(d^).
For each iteration in middle, detector gives a L'(d^) value input to the siso decoder, which outputs L_e(d) updates this to be the next L(d) gives to the detector. The detector then again updates L'(d^) and same procedure repeats until convergence is seen and final L(d^) i.e. output LLR value is achieved.

Log - likelihood algebra

It's all XOR!!

likelihood algebra [5] is introduced. For statistically independent data d, the sum of two log-likelihood ratios (LLRs) is defined as follows:

$$L(d_1) \boxplus L(d_2) \triangleq L(d_1 \oplus d_2) = \log_e\left[\frac{e^{L(d_1)} + e^{L(d_2)}}{1 + e^{L(d_1)}\, e^{L(d_2)}}\right] \tag{12}$$

$$\approx (-1) \times \operatorname{sgn}[L(d_1)] \times \operatorname{sgn}[L(d_2)] \times \min(|L(d_1)|, |L(d_2)|) \tag{13}$$

There are 3 pluses. The square one is log likelihood addition between two LLRs defines using the circle one. That expression basically says, "Take LLR of XOR of d_1 and d_2". If d_1=d_2= 0 or 1, xor= 0, if exactly one of them is zero, XOR is 1.

It is mathematically defined by the rightmost of equation 12 and mathematically approximated using equation 13. Note and understand how the sign of L(d1) and L(d2) rightly contribute using signum function and together with minus sign represent XOR of d1 and d2.
For practice, see that:

$$L(d) \boxplus \infty = -L(d)$$

$$L(d) \boxplus 0 = 0$$