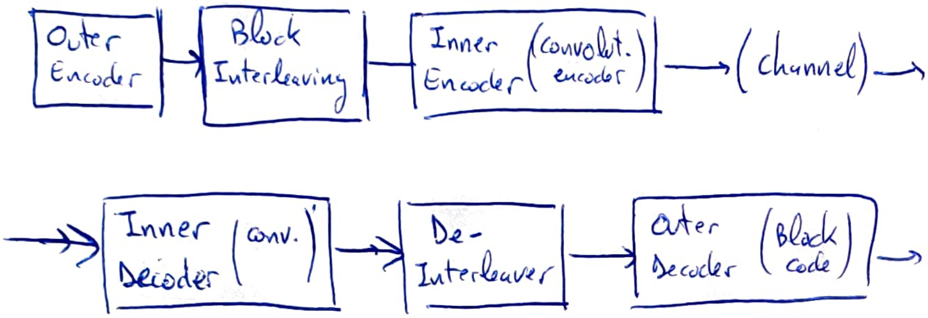
Lecture 8 : Turbo Codes

Classic Concatenated Coding

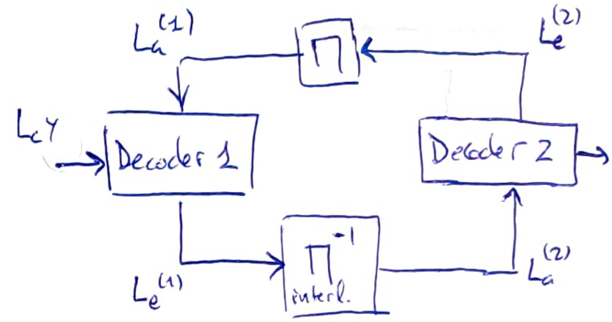
It was a series concatenation



Principal Characteristics

* Parallel, instead of series, concatenated coding
* Use of convolutional codes
* Pseudo-random interleaving
* Iterative (SISO) decoding → soft in, soft out decoding

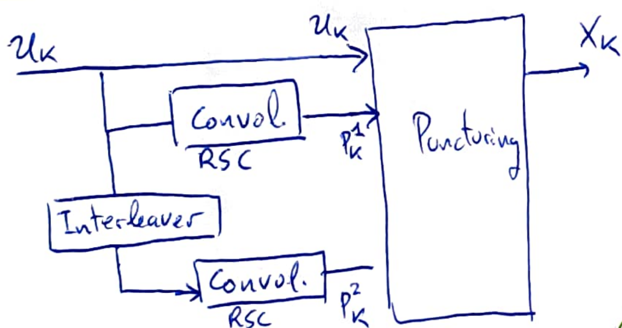
Turbo Principle



The of turbo decoder

* We have a first decoder and it gives a first output and the after the interleaving it goes to a second decoder
* Instead of deciding the final result, we send it to the first decoder, completing in this way one iteration.
* Then, the first decoder receives a "cleaned" information (cleaned by 2nd decoder) We continue doing iterations.
* Where nothing changes from one iteration to another the process stops and it produces the final result.

Turbo Encoder Architecture



: u_k is the original information and it is send to the channel in different ways

① It is send directly

② The same u_k is send to a first convolutional encoder (Systematic Conv. Encod.) which produces the parity check bits P_k^1 that are send

③ The same u_k is send to an interleaver (a mix in time domain between bits) and to another convolutional encoder that produces the second parity check P_k^2 that is also send.

* RSC → Recursive Systematic Convolutional – encoders

④ with what we have described the rate will be very long so there is a puncturing where we discard (with precise procedure) some bits

⑤ The final result is send on the channel.

SISO decoder

Soft in - soft out ⇒ We are working with the real values taken by the sampler

* We work with logaritms because is more convenient

* Inputs/output of SISO Decoder

↳ Log likehood ratio: It is the log. of the probability that the value assume +1 divided by the prob. of the value assume −1

$$L(u) = \log\left(\frac{P(u=+1)}{P(u=-1)}\right) \Rightarrow \begin{cases} L(u) > 0 \Rightarrow \dfrac{P(u=+1)}{P(u=-1)} > 1 \\ L(u) < 0 \Rightarrow \dfrac{P(u=+1)}{P(u=-1)} < 1 \end{cases}$$

The sign of $L(u)$ decide who is the winner.

↳ At the input we have

$L_c(u)$ → log likehood ratio associated to the original information

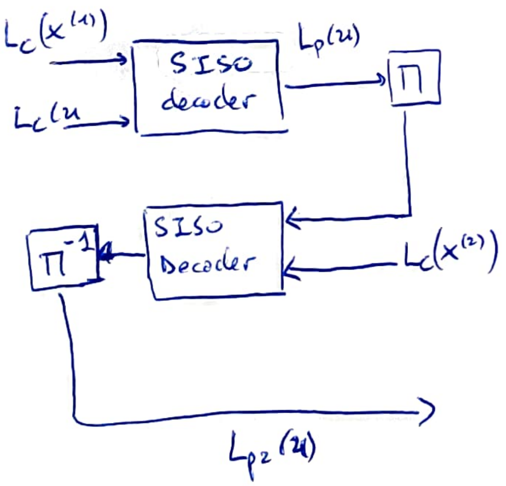
$L_c(x^{(1)})$ → " " " " to the parity check control

↳ Output: It produces the posteriori value related to the original information

Using BCJR algorithm evaluate the max. a posteriori prob.

$L_p(u)$ → This info. is more affordable than $L_c(u)$

* Decoding



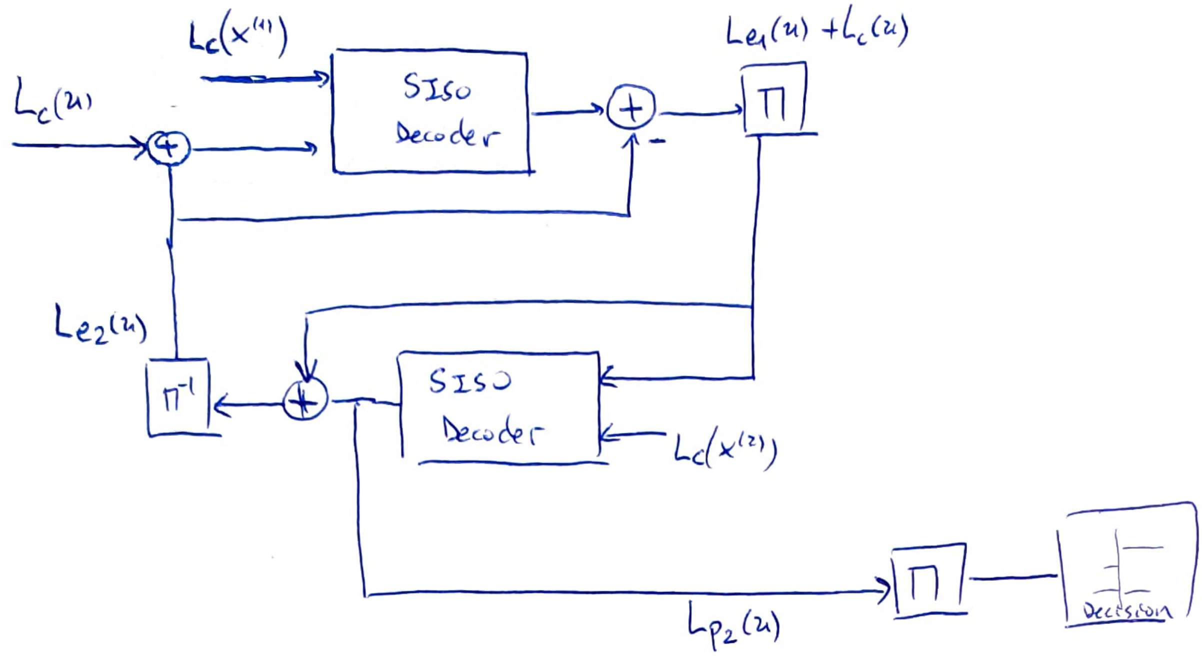
↳ In the first SISO decoder we apply channel transmitted information $L_c(u)$ and the channel information related to the first parity check control. $L_c(x^{(1)})$
Apply the procedure and obtain a posteriori info related to the original data $L_p(u)$

↳ Apply the permutation and give the value to the second siso dec. which uses the second parity check control

↳ Rearrange the bit in the proper order

↳ We have the output $L_{p2}(u)$ so we can decide the final result

* Architecture of iterative decoding



* Here we give to the first siso dec. the information given by th channel $L_c(u)$ and the info. of the channel related to the first parity check control $L_c(x^{(1)})$.
* We do the permutation and give to the second siso dec.
* Instead of deciding inmediatly we come back to the feedback loop.

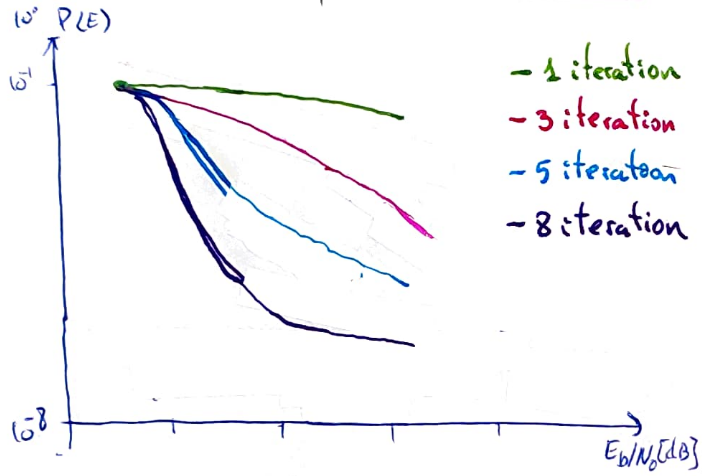
⇒ Important: To avoid positive feedback we have to remove the information that was already given. ⇒ Extrinsic information

⇒ Extrinsic information:
- The extrinsic info is found by substructing the corresponding input from the output
- It is necessary to subtract the info. that is already available at the decoder to prevent positive feedback
- The extrinsic info is the amount of new info gained by the current decoder step

Performance

* According to Shannon the requirements to have a good performance
  - ↳ Random codes and large blocks ⇒ Many bits
  - ↳ Soft Decoding ⇒ Turbo uses convolutional codes (soft decoding can be implemented using Vitrebi algorithm)

* the behaviour respect the iterations



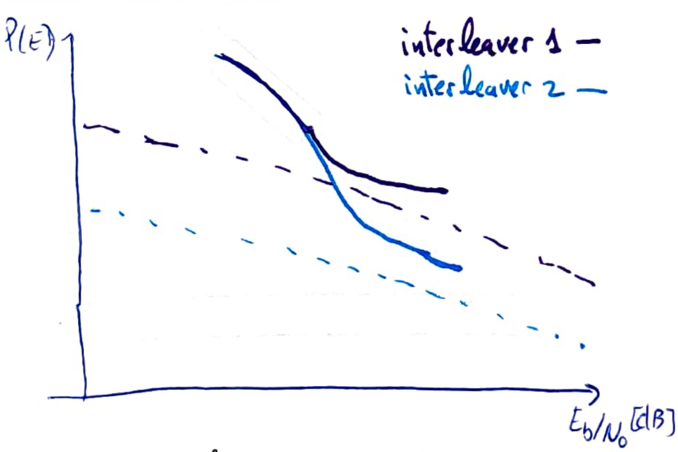
We can see that the behaviour of P(E) is composed by two parts: Waterfall (cliff) and Error Floor

→ The first part is the waterfall and we can see that increasing the SNR we decrease the P(E) a lot

→ The second part is the Error Floor and we can see that even if we increase SNR the improvement of the performance is not very important.

Finally, we see that increasing the iterations leads to a better performances, besides we don't need many iterations to reach to a very good performance.

* The role of the Interleaver



We can see here that if we change the interleaver, the floor will be different

During the the waterfall the interleaver is not important, it only affects strongly the performance of the floor

The interleaver is related to the permutation ⇒ but is not clear what is the best permutation to be applied

* Random coding argument:

- Truly random codes approaches capacity ⇒ but not feasible
- Turbo codes appear random ⇒ but it allows practical decoding

* Distance spectrum argument

- Traditionally the code design is focused on ⇒ Maximizing Minimum Distance
- With Turbo Codes we have a minimum distance that is not so big

  the goal is: ⇒ Reduce the multiplicity of low weight code words

  ↳ when I transmit a code word there are many other codewords that are close to the transmitted one ⇒ We try to reduce it

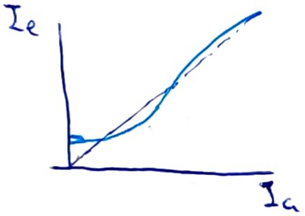
↳ So with smal d_{min} a remarkable performance can be achieve at low SNR

EXIT Chart (EXtrinsic Information Transfer)

* It is a way to analyze the performance and the behaviour of a Turbo Codes with respect the iterations

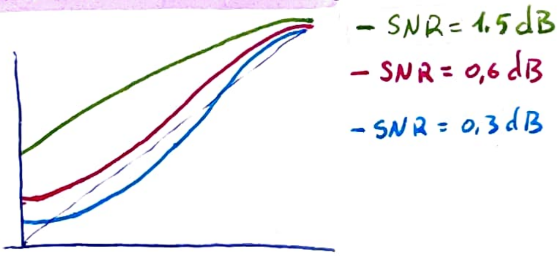
* I_a axis: Shows the mutual information at the input of the SISO Decoder
The mutual information between the original data and the information given in the input of SISO dec.

* I_e axis: Shows the mutual information at the output of SISO Decoder.
Mutual info between output and the original information.



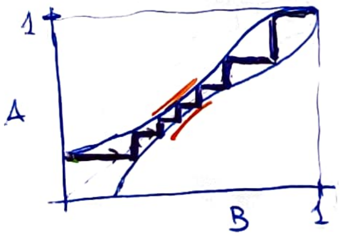
If we increase I_a (mutual info at input), the I_e (mutual info at output) will be also increased

* Behaviour with SNR

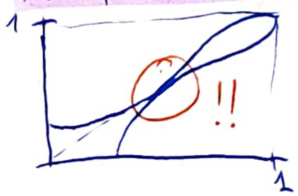


Increasing the SNR I will obtain better performances
The mutual info. at output is increased when SNR is increased because the noise will be less important

* Analysis of Turbo Codes



A → output of 1st decoder becomes the input to 2nd decoder
B → Output of 2nd decoder becomes the input to 1st decoder



We are working with the same code for the two decoders
The output of the first dec. becomes the input of the second dec. and then the output of the second code becomes the input of the first dec.. This repeats iteratively.

⇒ The path shows us what happens with the iterations between one code and the other

⇒ We use the EXIT chart in this way to analize the convergence

⇒ When the part named the tunnel is reduced we need more iterations to pass through it
If the tunnel is closed the convergence is not guaranteed

The goal of the iterations is to reach to the mutual info equal to 1.
The EXIT chart is the best way to analyze the behaviour of the code with respect iterations and final goal