# An introduction to coding theory

Adrish Banerjee

Department of Electrical Engineering
Indian Institute of Technology Kanpur
Kanpur, Uttar Pradesh
India

Mar. 6, 2017

## Lecture #16A: Turbo Decoding

# Concatenated codes

## Outline of the lecture

- Review of BCJR algorithm in log domain.

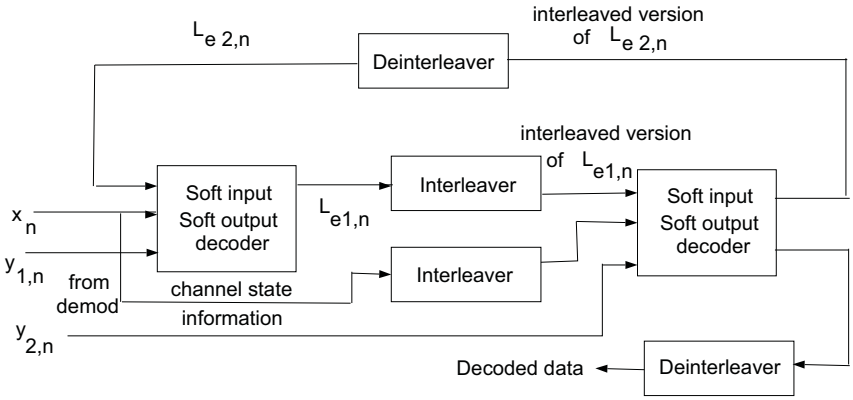


# Concatenated codes

## Outline of the lecture

- Review of BCJR algorithm in log domain.
- Turbo decoding for rate $R = 1/3$ code.

# Turbo decoding






# BCJR Algorithm

- Define $max^*(\cdot)$ function:

$$max^*(x, y) \equiv \ln(e^x + e^y) = max(x, y) + \ln(1 + e^{-|x-y|})$$

$$max^*(x, y, z) \equiv \ln(e^x + e^y + e^z) = max^*[max^*(x, y), z],$$

## BCJR Algorithm

- Define $max^*(\cdot)$ function:

$$max^*(x,y) \equiv \ln(e^x + e^y) = max(x,y) + \ln(1 + e^{-|x-y|})$$

$$max^*(x,y,z) \equiv \ln(e^x + e^y + e^z) = max^*[max^*(x,y),z],$$

- Branch metrics:

$$\gamma_l^*(s',s) \equiv \ln\ \gamma_l(s',s) = \frac{u_l L_a(u_l)}{2} + \frac{L_c}{2}\mathbf{r}_l \cdot \mathbf{v}_l$$



## BCJR Algorithm

- Define $max^*(\cdot)$ function:

$$max^*(x,y) \equiv \ln(e^x + e^y) = max(x,y) + \ln(1 + e^{-|x-y|})$$

$$max^*(x,y,z) \equiv \ln(e^x + e^y + e^z) = max^*[max^*(x,y),z],$$

- Branch metrics:

$$\gamma_l^*(s',s) \equiv \ln\ \gamma_l(s',s) = \frac{u_l L_a(u_l)}{2} + \frac{L_c}{2}\mathbf{r}_l \cdot \mathbf{v}_l$$

- Forward metrics:

$$\alpha_{l+1}^*(s) \equiv \ln\ \alpha_{l+1}(s) = max^*_{s' \in \sigma_l}[\gamma_l^*(s',s) + \alpha_l^*(s')]$$

## BCJR Algorithm

- Define $max^*(\cdot)$ function:

$$max^*(x,y) \equiv \ln(e^x + e^y) = max(x,y) + \ln(1 + e^{-|x-y|})$$

$$max^*(x,y,z) \equiv \ln(e^x + e^y + e^z) = max^*[max^*(x,y), z],$$

- Branch metrics:

$$\gamma_l^*(s',s) \equiv \ln\ \gamma_l(s',s) = \frac{u_l L_a(u_l)}{2} + \frac{L_c}{2}\mathbf{r}_l \cdot \mathbf{v}_l$$

- Forward metrics:

$$\alpha_{l+1}^*(s) \equiv \ln\ \alpha_{l+1}(s) = max^*_{s' \in \sigma_l}[\gamma_l^*(s',s) + \alpha_l^*(s')]$$

- Forward metrics initialization:

$$\alpha_0^*(s) \equiv \ln\ \alpha_0(s) = \begin{cases} 0, & s = \mathbf{0} \\ -\infty, & s \neq \mathbf{0}, \end{cases}$$



## BCJR Algorithm

- Backward metrics:

$$\beta_l^*(s') \equiv \ln\ \beta_l(s') = max^*_{s \in \sigma_{l+1}}[\gamma_l^*(s',s) + \beta_{l+1}^*(s)],$$

## BCJR Algorithm

- Backward metrics:

$$\beta_l^*(s') \equiv \ln\ \beta_l(s') = max^*_{s\in\sigma_{l+1}}[\gamma_l^*(s',s) + \beta_{l+1}^*(s)],$$

- Backward metrics initialization:

$$\beta_K^*(s) \equiv \ln\ \beta_K(s) = \begin{cases} 0, & s = \mathbf{0} \\ -\infty, & s \neq \mathbf{0}. \end{cases}$$



## BCJR Algorithm

- Backward metrics:

$$\beta_l^*(s') \equiv \ln\ \beta_l(s') = max^*_{s\in\sigma_{l+1}}[\gamma_l^*(s',s) + \beta_{l+1}^*(s)],$$

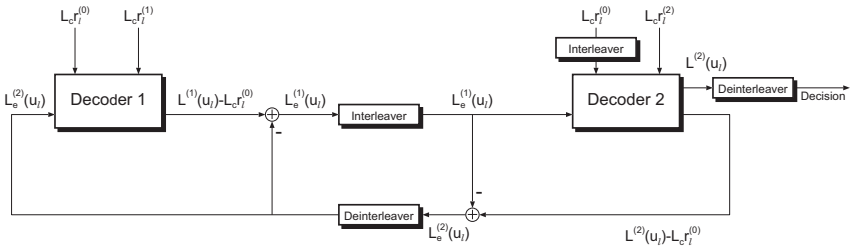
- Backward metrics initialization:

$$\beta_K^*(s) \equiv \ln\ \beta_K(s) = \begin{cases} 0, & s = \mathbf{0} \\ -\infty, & s \neq \mathbf{0}. \end{cases}$$

- APP L-value:

$$\begin{aligned} L(u_l) \quad = \quad & max^*_{(s',s)\in\Sigma_l^+}[\beta_{l+1}^*(s) + \gamma_l^*(s',s) + \alpha_l^*(s')] \\ & - max^*_{(s',s)\in\Sigma_l^-}[\beta_{l+1}^*(s) + \gamma_l^*(s',s) + \alpha_l^*(s')]. \end{aligned}$$

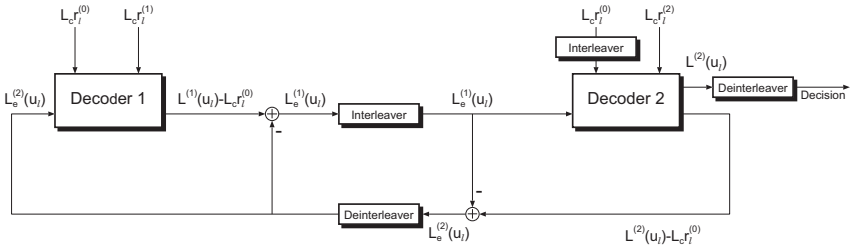
# Turbo decoding




- Basic structure of an iterative turbo decoder for a rate 1/3 turbo code.

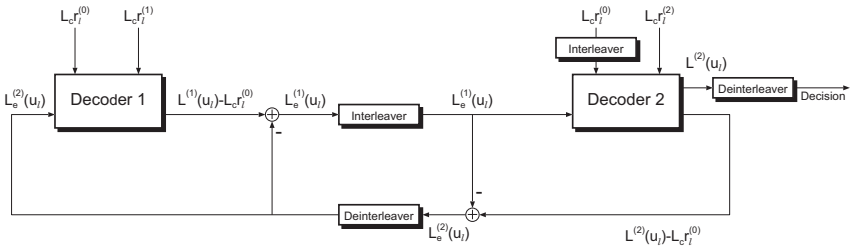


# Turbo decoding




- Basic structure of an iterative turbo decoder for a rate 1/3 turbo code.
- It employs two SISO decoders using the MAP algorithm.

# Turbo decoding




- Basic structure of an iterative turbo decoder for a rate $1/3$ turbo code.
- It employs two SISO decoders using the MAP algorithm.
- At each time unit $l$, three output values are received from the channel, one for the information bit $u_l = v_l^{(0)}$, denoted $r_l^{(0)}$, and two for the parity bits $v_l^{(1)}$ and $v_l^{(2)}$, denoted $r_l^{(1)}$ and $r_l^{(2)}$.



# Turbo decoding

- The $3K$-dimensional received vector is denoted by

$$\mathbf{r} = \left( r_0^{(0)} r_0^{(1)} r_0^{(2)}, r_1^{(0)} r_1^{(1)} r_1^{(2)}, \cdots, r_{K-1}^{(0)} r_{K-1}^{(1)} r_{K-1}^{(2)} \right).$$

- Assume 0 is mapped to $-1$ and 1 to $+1$.
- Then for an AWGN channel, we define the log-likelihood ratio (L-value) $L\left(u_l \mid r_l^{(0)}\right)$ (before decoding) of a transmitted information bit $u_l$ given the received value $r_l^{(0)}$ as

$$\begin{aligned} L(u_l \mid r_l^{(0)}) &= \ln \frac{P\left(u_l = +1 \mid r_l^{(0)}\right)}{P\left(u_l = -1 \mid r_l^{(0)}\right)} \\ &= \ln \frac{P\left(r_l^{(0)} \mid u_l = +1\right) P\left(u_l = +1\right)}{P\left(r_l^{(0)} \mid u_l = -1\right) P\left(u_l = -1\right)} \end{aligned}$$

## Turbo decoding

$$
\begin{aligned}
L(u_l \mid r_l^{(0)}) &= \ln \frac{P\left(r_l^{(0)} \mid u_l = +1\right)}{P\left(r_l^{(0)} \mid u_l = -1\right)} + \ln \frac{P(u_l = +1)}{P(u_l = -1)} \\
&= \ln \frac{e^{-(E_s/N_0)\left(r_l^{(0)} - 1\right)^2}}{e^{-(E_s/N_0)\left(r_l^{(0)} + 1\right)^2}} + \ln \frac{P(u_l = +1)}{P(u_l = -1)}, \\
&= -\frac{E_s}{N_0}\left\{\left(r_l^{(0)} - 1\right)^2 - \left(r_l^{(0)} + 1\right)^2\right\} + \ln \frac{P(u_l = +1)}{P(u_l = -1)} \\
&= 4\frac{E_s}{N_0} r_l^{(0)} + \ln \frac{P(u_l = +1)}{P(u_l = -1)} \\
&= L_c r_l^{(0)} + L_a(u_l),
\end{aligned}
$$

where $E_s/N_0$ is the channel SNR , $u_l$ and $r_l^{(0)}$ have both been normalized by a factor of $\sqrt{E_s}$, $L_c = 4(E_s/N_0)$ is the channel reliability factor and $L_a(u_l)$ is the a priori L-value of the bit $u_l$.



## Turbo decoding

- In the case of a transmitted parity bit $v_l^{(j)}$, given the received value $r_l^{(j)}$, $j = 1, 2$, the $L$-value (before decoding) is given by

$$
L\left(v_l^{(j)} \mid r_l^{(j)}\right) = L_c r_l^{(j)} + L_a\left(v_l^{(j)}\right) = L_c r_l^{(j)}, \quad j = 1, 2, \quad \text{(why ?)}
$$

## Turbo decoding

- In the case of a transmitted parity bit $v_l^{(j)}$, given the received value $r_l^{(j)}$, $j = 1, 2$, the $L$-value (before decoding) is given by

$$L\left(v_l^{(j)} \mid r_l^{(j)}\right) = L_c r_l^{(j)} + L_a\left(v_l^{(j)}\right) = L_c r_l^{(j)}, \quad j = 1, 2, \quad \text{(why ?)}$$

- $L_a(u_l)$ also equals 0 for the first iteration of decoder 1, but that thereafter the a priori $L$-values of the information bits are replaced by extrinsic $L$-values from the other decoder.



## Turbo decoding

- In the case of a transmitted parity bit $v_l^{(j)}$, given the received value $r_l^{(j)}$, $j = 1, 2$, the $L$-value (before decoding) is given by

$$L\left(v_l^{(j)} \mid r_l^{(j)}\right) = L_c r_l^{(j)} + L_a\left(v_l^{(j)}\right) = L_c r_l^{(j)}, \quad j = 1, 2, \quad \text{(why ?)}$$

- $L_a(u_l)$ also equals 0 for the first iteration of decoder 1, but that thereafter the a priori $L$-values of the information bits are replaced by extrinsic $L$-values from the other decoder.
- The received soft channel $L$-values $L_c r_l^{(0)}$ for $u_l$ and $L_c r_l^{(1)}$ for $v_l^{(1)}$ enter decoder 1, while the (properly interleaved) received soft channel $L$-values $L_c r_l^{(0)}$ for $u_l$ and the received soft channel $L$-values $L_c r_l^{(2)}$ for $v_l^{(2)}$ enter decoder 2.

# Turbo decoding

- The output of decoder 1 contains two terms:



# Turbo decoding

- The output of decoder 1 contains two terms:
  - $L^{(1)}(u_l) = \ln\left[P\left(u_l = +1 \;/\; \mathbf{r}_1, \mathbf{L}_a^{(1)}\right) \Big/ P\left(u_l = -1 \mid \mathbf{r}_1, \mathbf{L}_a^{(1)}\right)\right],$

## Turbo decoding

- The output of decoder 1 contains two terms:
  - $L^{(1)}(u_l) = \ln\left[P\left(u_l = +1 \,/\, \mathbf{r}_1, \mathbf{L}_a^{(1)}\right) \Big/ P\left(u_l = -1 \mid \mathbf{r}_1, \mathbf{L}_a^{(1)}\right)\right]$,
  - $L_e^{(1)}(u_l) = L^{(1)}(u_l) - \left[L_c r_l^{(0)} + L_e^{(2)}(u_l)\right]$,



## Turbo decoding

- The output of decoder 1 contains two terms:
  - $L^{(1)}(u_l) = \ln\left[P\left(u_l = +1 \,/\, \mathbf{r}_1, \mathbf{L}_a^{(1)}\right) \Big/ P\left(u_l = -1 \mid \mathbf{r}_1, \mathbf{L}_a^{(1)}\right)\right]$,
  - $L_e^{(1)}(u_l) = L^{(1)}(u_l) - \left[L_c r_l^{(0)} + L_e^{(2)}(u_l)\right]$,
- $L^{(1)}(u_l)$ is the a posteriori $L$-value (after decoding) of each information bit produced by decoder 1 given the (partial) received vector $\mathbf{r}_1 \stackrel{\Delta}{=} \left[r_0^{(0)} r_0^{(1)}, r_1^{(0)} r_1^{(1)}, \cdots, r_{K-1}^{(0)} r_{K-1}^{(1)}\right]$ and the a priori input vector $\mathbf{L}_a^{(1)} \stackrel{\Delta}{=} \left[L_a^{(1)}(u_0), L_a^{(1)}(u_1), \cdots, L_a^{(1)}(u_{K-1})\right]$ for decoder 1.

## Turbo decoding

- The output of decoder 1 contains two terms:
  - $L^{(1)}(u_l) = \ln\left[P\left(u_l = +1 \,/\, \mathbf{r}_1, \mathbf{L}_a^{(1)}\right) \Big/ P\left(u_l = -1 \mid \mathbf{r}_1, \mathbf{L}_a^{(1)}\right)\right]$,
  - $L_e^{(1)}(u_l) = L^{(1)}(u_l) - \left[L_c r_l^{(0)} + L_e^{(2)}(u_l)\right]$,
- $L^{(1)}(u_l)$ is the a posteriori $L$-value (after decoding) of each information bit produced by decoder 1 given the (partial) received vector $\mathbf{r}_1 \stackrel{\Delta}{=} \left[r_0^{(0)} r_0^{(1)}, r_1^{(0)} r_1^{(1)}, \cdots, r_{K-1}^{(0)} r_{K-1}^{(1)}\right]$ and the a priori input vector $\mathbf{L}_a^{(1)} \stackrel{\Delta}{=} \left[L_a^{(1)}(u_0), L_a^{(1)}(u_1), \cdots, L_a^{(1)}(u_{K-1})\right]$ for decoder 1.
- $L_e^{(1)}(u_l)$ is the extrinsic a posteriori $L$-value (after decoding) associated with each information bit produced by decoder 1, which, after interleaving, is passed to the input of decoder 2 as the a priori value $L_a^{(2)}(u_l)$.



## Turbo decoding

- Subtracting the terms, viz., $L_c r_l^{(0)} + L_e^{(2)}(u_l)$ from $L^{(1)}(u_l)$, removes the effect of the current information bit $u_l$ from $L^{(1)}(u_l)$, and thus providing an independent estimate of the information bit $u_l$ to decoder 2 in addition to the received soft channel $L$-values at time $l$.

## Turbo decoding

- Subtracting the terms, viz., $L_c r_l^{(0)} + L_e^{(2)}(u_l)$ from $L^{(1)}(u_l)$, removes the effect of the current information bit $u_l$ from $L^{(1)}(u_l)$, and thus providing an independent estimate of the information bit $u_l$ to decoder 2 in addition to the received soft channel $L$-values at time $l$.
- Similarly, the output of decoder 2 contains two terms:



## Turbo decoding

- Subtracting the terms, viz., $L_c r_l^{(0)} + L_e^{(2)}(u_l)$ from $L^{(1)}(u_l)$, removes the effect of the current information bit $u_l$ from $L^{(1)}(u_l)$, and thus providing an independent estimate of the information bit $u_l$ to decoder 2 in addition to the received soft channel $L$-values at time $l$.
- Similarly, the output of decoder 2 contains two terms:
  - $L^{(2)}(u_l) = \ln\left[P\left(u_l = +1 \;/\; \mathbf{r}_2, \mathbf{L}_a^{(2)}\right) \Big/ P\left(u_l = -1 \mid \mathbf{r}_2, \mathbf{L}_a^{(2)}\right)\right]$,

    where $\mathbf{r}_2$ is the (partial) received vector and $\mathbf{L}_a^{(2)}$ the a priori input vector for decoder 2, and

## Turbo decoding

- Subtracting the terms, viz., $L_c r_l^{(0)} + L_e^{(2)}(u_l)$ from $L^{(1)}(u_l)$, removes the effect of the current information bit $u_l$ from $L^{(1)}(u_l)$, and thus providing an independent estimate of the information bit $u_l$ to decoder 2 in addition to the received soft channel $L$-values at time $l$.
- Similarly, the output of decoder 2 contains two terms:
  - $L^{(2)}(u_l) = \ln \left[ P\left(u_l = +1 \;/\; \mathbf{r}_2, \mathbf{L}_a^{(2)}\right) \Big/ P\left(u_l = -1 \mid \mathbf{r}_2, \mathbf{L}_a^{(2)}\right) \right]$, where $\mathbf{r}_2$ is the (partial) received vector and $\mathbf{L}_a^{(2)}$ the a priori input vector for decoder 2, and
  - $L_e^{(2)}(u_l) = L^{(2)}(u_l) - \left[ L_c r_l^{(0)} + L_e^{(1)}(u_l) \right]$, and the extrinsic *a posteriori* $L$-values $L_e^{(2)}(u_l)$ produced by decoder 2, after deinterleaving, are passed back to the input of decoder 1 as the a priori values $L_a^{(1)}(u_l)$.



## Turbo decoding

- Thus, the input to each decoder contains three terms,

# Turbo decoding

- Thus, the input to each decoder contains three terms,
  - The soft channel $L$-values $L_c r_l^{(0)}$ (information bit).



# Turbo decoding

- Thus, the input to each decoder contains three terms,
  - The soft channel $L$-values $L_c r_l^{(0)}$ (information bit).
  - The soft channel $L$-values$L_c r_l^{(1)}$ (or $L_c r_l^{(2)}$) (parity bit),

# Turbo decoding

- Thus, the input to each decoder contains three terms,
  - The soft channel $L$-values $L_c r_l^{(0)}$ (information bit).
  - The soft channel $L$-values$L_c r_l^{(1)}$ (or $L_c r_l^{(2)}$) (parity bit),
  - The extrinsic a posteriori $L$-values $L_e^{(2)}(u_l) = L_a^{(1)}(u_l)$ (or $L_e^{(1)}(u_l) = L_a^{(2)}(u_l)$) passed from the other decoder.



# Turbo decoding

- Thus, the input to each decoder contains three terms,
  - The soft channel $L$-values $L_c r_l^{(0)}$ (information bit).
  - The soft channel $L$-values$L_c r_l^{(1)}$ (or $L_c r_l^{(2)}$) (parity bit),
  - The extrinsic a posteriori $L$-values $L_e^{(2)}(u_l) = L_a^{(1)}(u_l)$ (or $L_e^{(1)}(u_l) = L_a^{(2)}(u_l)$) passed from the other decoder.
- In the initial iteration of decoder 1, the extrinsic a posteriori $L$-values $L_e^{(2)}(u_l) = L_a^{(1)}(u_l)$ are just the original a priori $L$-values $L_a(u_l)$, which are all equal to 0 for equally likely information bits.

## Turbo decoding

- Thus, the input to each decoder contains three terms,
  - The soft channel $L$-values $L_c r_l^{(0)}$ (information bit).
  - The soft channel $L$-values$L_c r_l^{(1)}$ (or $L_c r_l^{(2)}$) (parity bit),
  - The extrinsic a posteriori $L$-values $L_e^{(2)}(u_l) = L_a^{(1)}(u_l)$ (or $L_e^{(1)}(u_l) = L_a^{(2)}(u_l)$) passed from the other decoder.
- In the initial iteration of decoder 1, the extrinsic a posteriori $L$-values $L_e^{(2)}(u_l) = L_a^{(1)}(u_l)$ are just the original a priori $L$-values $L_a(u_l)$, which are all equal to 0 for equally likely information bits.
- Thus the extrinsic $L$-values passed from one decoder to the other during the iterative decoding process are treated like new sets of *a priori* probabilities by the MAP algorithm.



## Turbo decoding

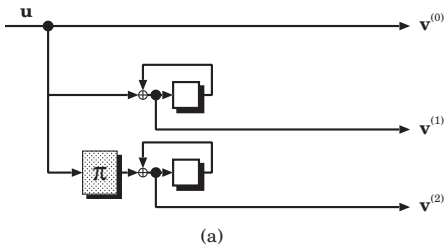
- Decoding then proceeds iteratively, with each decoder passing its respective extrinsic $L$-values back to the other decoder.
- Decoding stops after sufficient number of iterations.
- At the output of decoder 2, the decoded information bits are determined from the a posteriori L-values $L^{(2)}(u_l)$.
- Positive L-values are decoded as "+1" and negative L-values as "$-1$".

Example:

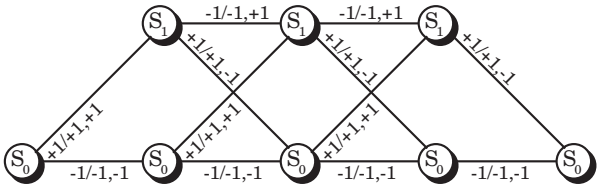
- Rate $R = 1/3$ turbo code using constituent encoder $\mathbf{G}(D) = \left[1 \quad \frac{1}{1+D}\right]$

(See Figure in the next frame.)

# Turbo decoding




(a)




(b)



# Turbo decoding

- Consider an input sequence of length $K = 4$, including one termination bit, along with a $2 \times 2$ block (row-column) interleaver, resulting in a $(12, 3)$ turbo code with overall rate $R = 1/4$.

## Turbo decoding

- Consider an input sequence of length $K = 4$, including one termination bit, along with a $2 \times 2$ block (row-column) interleaver, resulting in a $(12, 3)$ turbo code with overall rate $R = 1/4$.
- The trellis for the constituent code is shown in the previous frame, where the branches are labeled using the mapping $0 \rightarrow -1$ and $1 \rightarrow +1$.



## Turbo decoding

- Consider an input sequence of length $K = 4$, including one termination bit, along with a $2 \times 2$ block (row-column) interleaver, resulting in a $(12, 3)$ turbo code with overall rate $R = 1/4$.
- The trellis for the constituent code is shown in the previous frame, where the branches are labeled using the mapping $0 \rightarrow -1$ and $1 \rightarrow +1$.
- The input block is given by the vector $\mathbf{u} = [u_0, u_1, u_2, u_3]$, the interleaved input block is $\mathbf{u}' = [u'_0, u'_1, u'_2, u'_3] = [u_0, u_2, u_1, u_3]$,

## Turbo decoding

- Consider an input sequence of length $K = 4$, including one termination bit, along with a $2 \times 2$ block (row-column) interleaver, resulting in a $(12, 3)$ turbo code with overall rate $R = 1/4$.
- The trellis for the constituent code is shown in the previous frame, where the branches are labeled using the mapping $0 \rightarrow -1$ and $1 \rightarrow +1$.
- The input block is given by the vector $\mathbf{u} = [u_0, u_1, u_2, u_3]$, the interleaved input block is $\mathbf{u}' = [u'_0, u'_1, u'_2, u'_3] = [u_0, u_2, u_1, u_3]$,
- The parity vector for the first constituent code is given by $\mathbf{p}^{(1)} = \left[p_0^{(1)}, p_1^{(1)}, p_2^{(1)}, p_3^{(1)}\right]$.



## Turbo decoding

- Consider an input sequence of length $K = 4$, including one termination bit, along with a $2 \times 2$ block (row-column) interleaver, resulting in a $(12, 3)$ turbo code with overall rate $R = 1/4$.
- The trellis for the constituent code is shown in the previous frame, where the branches are labeled using the mapping $0 \rightarrow -1$ and $1 \rightarrow +1$.
- The input block is given by the vector $\mathbf{u} = [u_0, u_1, u_2, u_3]$, the interleaved input block is $\mathbf{u}' = [u'_0, u'_1, u'_2, u'_3] = [u_0, u_2, u_1, u_3]$,
- The parity vector for the first constituent code is given by $\mathbf{p}^{(1)} = \left[p_0^{(1)}, p_1^{(1)}, p_2^{(1)}, p_3^{(1)}\right]$.
- The parity vector for the second constituent code is $\mathbf{p}^{(2)} = \left[p_0^{(2)}, p_1^{(2)}, p_2^{(2)}, p_3^{(2)}\right]$.

## Turbo decoding

- The 12 transmitted bits are represented in a rectangular array, as shown in Figure in the next frame, where the input vector $\mathbf{u}$ determines the parity vector $\mathbf{p}^{(1)}$ in the first two rows and the interleaved input vector $\mathbf{u}'$ determines the parity vector $\mathbf{p}^{(2)}$ in the first two columns.



## Turbo decoding

- The 12 transmitted bits are represented in a rectangular array, as shown in Figure in the next frame, where the input vector $\mathbf{u}$ determines the parity vector $\mathbf{p}^{(1)}$ in the first two rows and the interleaved input vector $\mathbf{u}'$ determines the parity vector $\mathbf{p}^{(2)}$ in the first two columns.
- For purposes of illustration, we assume the particular bit values shown in Figure.

# Turbo decoding

- The 12 transmitted bits are represented in a rectangular array, as shown in Figure in the next frame, where the input vector $\mathbf{u}$ determines the parity vector $\mathbf{p}^{(1)}$ in the first two rows and the interleaved input vector $\mathbf{u}'$ determines the parity vector $\mathbf{p}^{(2)}$ in the first two columns.
- For purposes of illustration, we assume the particular bit values shown in Figure.
- We also assume a channel SNR of $E_s/N_0 = 1/4$ $(-6.02\text{dB})$, so that the received channel $L$-values corresponding to the received vector $\mathbf{r} = \left[r_0^{(0)} r_0^{(1)} r_0^{(2)}, r_1^{(0)} r_1^{(1)} r_1^{(2)}, r_2^{(0)} r_2^{(1)} r_2^{(2)}, r_3^{(0)} r_3^{(1)} r_3^{(2)}\right]$ are given by

$$L_c r_l^{(j)} = 4\left(\frac{E_s}{N_0}\right) r_l^{(j)} = r_l^{(j)}, \quad l = 0, 1, 2, 3, \quad j = 0, 1, 2. \qquad (1)$$



# Turbo decoding

| | | | |
|---|---|---|---|
| $u_0$ | $u_1$ | $p_0^{(1)}$ | $p_1^{(1)}$ |
| $u_2$ | $u_3$ | $p_2^{(1)}$ | $p_3^{(1)}$ |
| $p_0^{(2)}$ | $p_2^{(2)}$ | | |
| $p_1^{(2)}$ | $p_3^{(2)}$ | | |

(12, 3) PCCC

| | | | |
|---|---|---|---|
| −1 | +1 | −1 | +1 |
| −1 | +1 | +1 | −1 |
| −1 | +1 | | |
| −1 | −1 | | |

Coded Values

| | | | |
|---|---|---|---|
| + 0.8 | + 1.0 | + 0.1 | − 0.5 |
| − 1.8 | + 1.6 | + 1.1 | − 1.6 |
| − 1.2 | + 0.2 | | |
| +1.2 | −1.1 | | |

Received L−values

## Turbo decoding

- In the first iteration of decoder 1 (row decoding), the BCJR algorithm is applied to the trellis of the 2-state $(2, 1, 1)$ code to compute the a posteriori $L$-values $L^{(1)}(u_l)$ for each of the four input bits and the corresponding extrinsic a posteriori $L$-values $L_e^{(1)}(u_l)$.



## Turbo decoding

- In the first iteration of decoder 1 (row decoding), the BCJR algorithm is applied to the trellis of the 2-state $(2, 1, 1)$ code to compute the a posteriori $L$-values $L^{(1)}(u_l)$ for each of the four input bits and the corresponding extrinsic a posteriori $L$-values $L_e^{(1)}(u_l)$.
- For iterative decoding, extrinsic a-posteriori L-values are computed for all input bits, termination bits as well as information bits.

## Turbo decoding

- In the first iteration of decoder 1 (row decoding), the BCJR algorithm is applied to the trellis of the 2-state $(2, 1, 1)$ code to compute the a posteriori $L$-values $L^{(1)}(u_l)$ for each of the four input bits and the corresponding extrinsic a posteriori $L$-values $L_e^{(1)}(u_l)$.
- For iterative decoding, extrinsic a-posteriori L-values are computed for all input bits, termination bits as well as information bits.
- Before the first iteration of decoding, the a-priori L-values of the termination bits are assumed to be zero.



## Turbo decoding

- In the first iteration of decoder 1 (row decoding), the BCJR algorithm is applied to the trellis of the 2-state $(2, 1, 1)$ code to compute the a posteriori $L$-values $L^{(1)}(u_l)$ for each of the four input bits and the corresponding extrinsic a posteriori $L$-values $L_e^{(1)}(u_l)$.
- For iterative decoding, extrinsic a-posteriori L-values are computed for all input bits, termination bits as well as information bits.
- Before the first iteration of decoding, the a-priori L-values of the termination bits are assumed to be zero.
- Similarly, in the first iteration of decoder 2, the BCJR algorithm uses the extrinsic a posteriori $L$-values $L_e^{(1)}(u_l)$ received from decoder 1 as the a priori $L$-values, $L_a^{(2)}(u_l)$ to compute the a posteriori $L$-values $L^{(2)}(u_l)$ for each of the four input bits and the corresponding extrinsic a posteriori $L$-values $L_e^{(2)}(u_l)$ to pass back to decoder 1.

# Turbo decoding

- In the first iteration of decoder 1 (row decoding), the BCJR algorithm is applied to the trellis of the 2-state $(2, 1, 1)$ code to compute the a posteriori $L$-values $L^{(1)}(u_l)$ for each of the four input bits and the corresponding extrinsic a posteriori $L$-values $L_e^{(1)}(u_l)$.
- For iterative decoding, extrinsic a-posteriori L-values are computed for all input bits, termination bits as well as information bits.
- Before the first iteration of decoding, the a-priori L-values of the termination bits are assumed to be zero.
- Similarly, in the first iteration of decoder 2, the BCJR algorithm uses the extrinsic a posteriori $L$-values $L_e^{(1)}(u_l)$ received from decoder 1 as the a priori $L$-values, $L_a^{(2)}(u_l)$ to compute the a posteriori $L$-values $L^{(2)}(u_l)$ for each of the four input bits and the corresponding extrinsic a posteriori $L$-values $L_e^{(2)}(u_l)$ to pass back to decoder 1.
- Decoding proceeds iteratively in this fashion.



# Turbo decoding

| | |
|---|---|
| −0.32 | −0.38 |
| +0.77 | +0.47 |

Extrinsic L−values after first row decoding

| | |
|---|---|
| −0.88 | −0.69 |
| +0.23 | −0.04 |

Extrinsic L−values after first column decoding

| | |
|---|---|
| −0.40 | −0.07 |
| −0.80 | +2.03 |

Soft−output L−values after the first row and column decoding

| | |
|---|---|
| −0.01 | −0.01 |
| +0.43 | +0.77 |

Extrinsic L−values after second row decoding

| | |
|---|---|
| −0.98 | −0.81 |
| +0.07 | −0.21 |

Extrinsic L−values after second column decoding

| | |
|---|---|
| −0.19 | +0.18 |
| −1.30 | +2.16 |

Soft−output L−values after the second row and column decoding