Channel Coding 2

Dr.-Ing. Dirk Wübben
Institute for Telecommunications and High-Frequency Techniques
Department of Communications Engineering
Room: N2300, Phone: 0421/218-62385
wuebben@ant.uni-bremen.de

Lecture

Tuesday, 08:30 – 10:00 in N3130

Exercise

Wednesday, 14:00 – 16:00 in N2420

Dates for exercises will be announced during lectures.

Tutor

Shayan Hassanpour

Room: N2390

Phone 218-62387

hassanpour@ant.uni-bremen.de

www.ant.uni-bremen.de/courses/cc2/

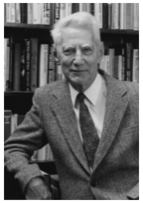
Outline Channel Coding II

- 1. Concatenated Codes
  - Serial Concatenation & Parallel Concatenation (Turbo Codes)
  - Iterative Decoding with Soft-In/Soft-Out decoding algorithms
  - EXIT-Charts
  - BiCM
  - LDPC Codes
- 2. Trelliscoded Modulation (TCM)
  - Motivation by information theory
  - TCM of Ungerböck, pragmatic approach by Viterbi, Multilevel codes
  - Distance properties and error rate performance
  - Applications (data transmission via modems)
- 3. Adaptive Error Control
  - Automatic Repeat Request (ARQ)
  - Performance for perfect and disturbed feedback channel
  - Hybrid FEC/ARQ schemes

Chapter 1. Concatenated Codes

- Introduction
  - Serial and Parallel Concatenation
- Interleaving
- Serial Concatenation
  - Direct approach, Product Codes, Choice of Component Codes
- Parallel Concatenation
  - Modification of Product Codes, Turbo-Codes, Choice of Component Codes
- Distance Properties and Performance Approximation
- Decoding of Concatenated Codes
  - Definition of Soft-Information, L-Algebra, General Approach for Soft-Output Decoding,
  - BCJR-Algorithm, Iterative Decoding, General Concept of Iterative Decoding
- EXtrinstic Information Transfer (EXIT)-Charts
- Bitinterleaved Coded Modulation (BiCM)
- Low Density Parity Check (LDPC) Codes

Introduction



Claude E. Shannon

David Forney

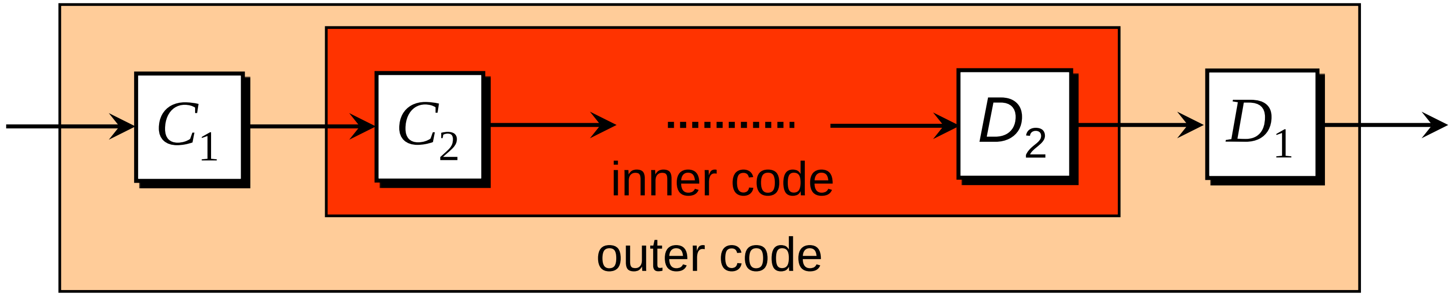
- Achieving Shannon’s channel capacity is the general goal of coding theory
- Block- and convolutional codes of CC-1 are far away from achieving this limit
  - Decoding effort increases (exponentially) with performance
  - Questionable, if Shannon’s limit can be achieved by these codes
- Concatenation of Codes
  - Forney (1966): proposed combination of simple codes
  - Berrou, Glaxieux, Thitimajshima: **Turbo-Codes** (1993):
    Clever parallel concatenation of two convolutional codes
    achieving 0.5 dB loss at $P_b=10^{-5}$ to channel capacity

Claude Berrou   Alain Glavieux   Punya Thitimajshima

- Principal Idea:
  - Clever concatenation of simple codes in order to generate a total code with high performance and enabling efficient decoding
  - Example:
    - Convolutional Code with $L_C = 9$ → $2^8 = 256$ states
    - 2 Convolutional Codes with $L_C = 3$ → $2 \cdot 2^2 = 8$ states → complexity reduction by a factor of 32
      repeated decoding (6 iterations) → $6 \cdot 8 = 48$ states → reduction by a factor of 5

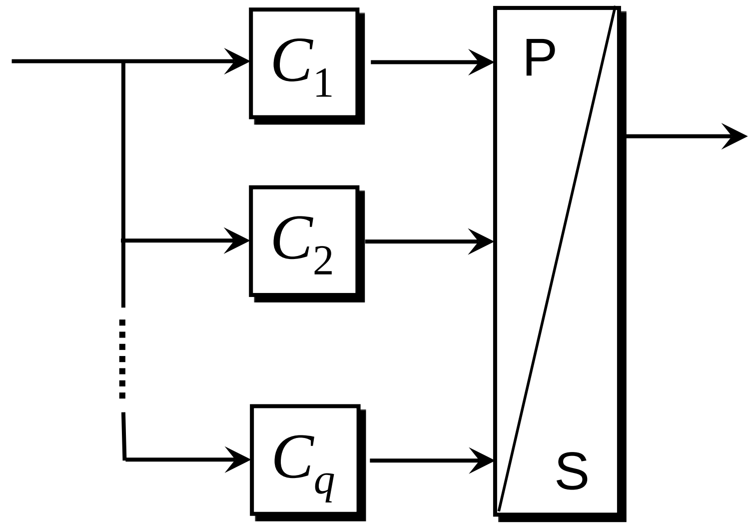
Serial and Parallel Code Concatenation

- **Serial Code Concatenation**

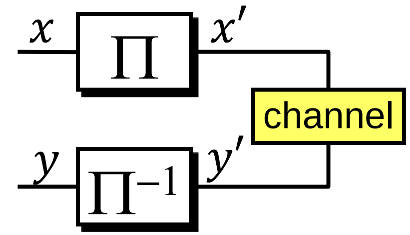


  - Subsequent encoder obtains whole output stream of previous encoder
    → redundancy bits are also encoded

- **Parallel Code Concatenation**
  - Each encoder obtains only information bits
  - Parallel-serial converter generates serial data stream
  - Example: Turbo Codes



Interleaving



- Interleaver performs permutation of symbol sequence
  - Strong impact on performance of concatenated codes
  - Also used to split burst errors into single errors for fading channels
- **Block interleaver**

write ↓   read →

x_0	x_3	x_6	x_9	x_{12}
x_1	x_4	x_7	x_{10}	x_{13}
x_2	x_5	x_8	x_{11}	x_{14}

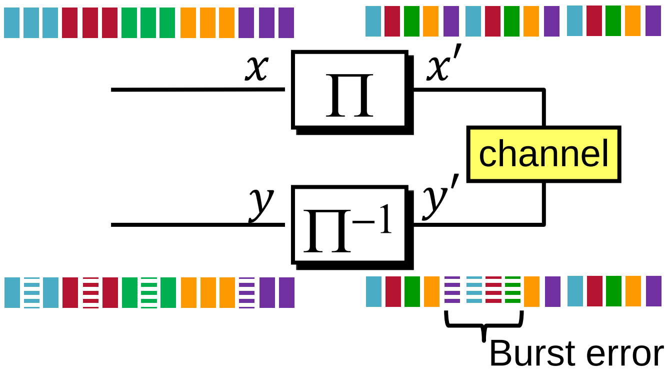
Column-wise write in, but row-wise read out leads to permutation of symbol sequence

interleaving depth $L_\mathrm{I} = 5$:
neighboring symbols of the input stream have a distance of 5 in the output stream
→ given by number of columns

  - input sequence: $x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}$
  - output sequence: $x_0, x_3, x_6, x_9, x_{12}, x_1, x_4, x_7, x_{10}, x_{13}, x_2, x_5, x_8, x_{11}, x_{14}$

L_I

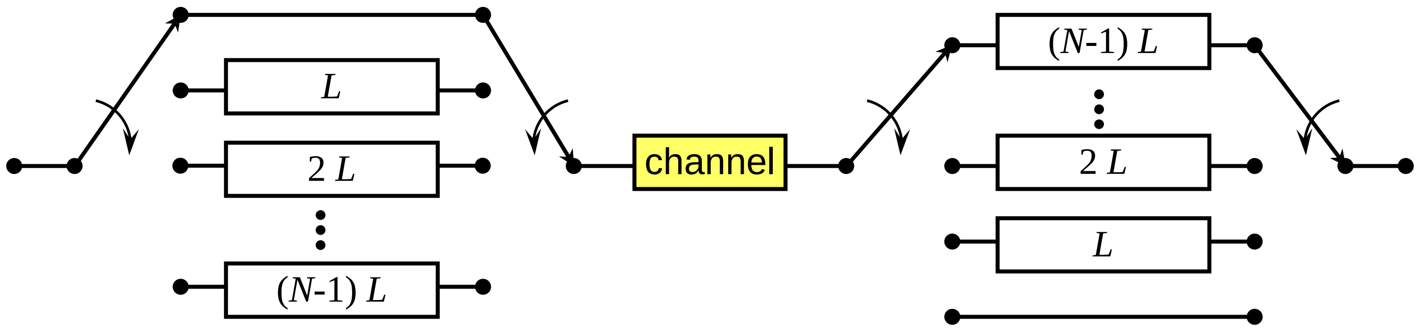
Interleaving



- Assumption: burst errors of length b should be separated
- Aspects of dimensioning block interleaver
  - Number of columns
    - affects directly the interleaver depth L_I
    - $L_\mathrm{I} \geq b$ is required, so that burst error of length b is broken into single errors by Π^{-1}
  - Number of rows
    - Example: For a convolutional code with $L_\mathrm{C} = 5$, five successive code words are correlated → for R_c =1/2 ten successive code bits are correlated
    - In order to separate these ten bits (by L_I to protect them from burst errors), the number of rows should correspond to $L_\mathrm{C}/R_\mathrm{c}$=10
  - Time delay (latency)
    - The memory is read out after the whole memory is written $\Delta t = \mathrm{rows} \cdot \mathrm{columns} \cdot T_b$
    - Notice: For duplex speech communication only an overall delay of 125 ms is tolerable
    - Example: data rate 9,6 kbit/s and interleaver size 400 bits $2 \cdot \Delta t = 2 \dfrac{400}{9600\,1/s} = 83{,}3\ \mathrm{ms}$

Interleaving

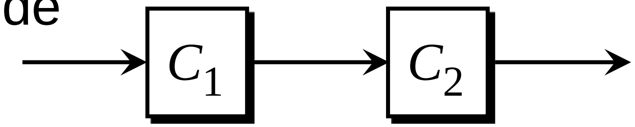
- **Convolutional Interleaver**



  - Consists of N registers and multiplexer
  - Each register stores L symbols more than the previous register
  - Principle is similar to block interleaver
- **Random Interleaver**
  - Block interleaver has a regular structure → output distance is directly given by input distance → leading to bad distance properties for Turbo-Codes
  - Random interleavers are constructed as block interleavers where the data positions are determined randomly
  - A **pseudo-random** generator can be utilized for constructing these interleavers

Serial Code Concatenation: Direct Approach

- Concatenation of (3,2,2)-SPC and (4,3,2)-SPC code



$\mathbf{u}$	$\mathbf{c}_1$	$\mathbf{c}_2$	$w_H(\mathbf{c}_2)$
00	000	0000	0
01	011	0110	2
10	101	1010	2
11	110	1100	2

$R_c = 2/4 = 1/2$

$d_{min} = 2$

Concatenation does not automatically result in a code with larger distance

- Concatenation of (4,3,2)-SPC and (7,4,3)-Hamming code

$\mathbf{u}$	$\mathbf{c}_1$	$\mathbf{c}_2$	$w_H(\mathbf{c}_2)$	$\mathbf{c}_2$	$w_H(\mathbf{c}_2)$
000	0000	0000 000	0	0000 000	0
001	0011	0011 001	3	0001 111	4
010	0101	0101 010	3	0110 011	4
011	0110	0110 011	4	0111 100	4
100	1001	1001 100	3	1010 101	4
101	1010	1010 101	4	1011 010	4
110	1100	1100 110	4	1100 110	4
111	1111	1111 111	7	1101 001	4

$R_c = 3/7$

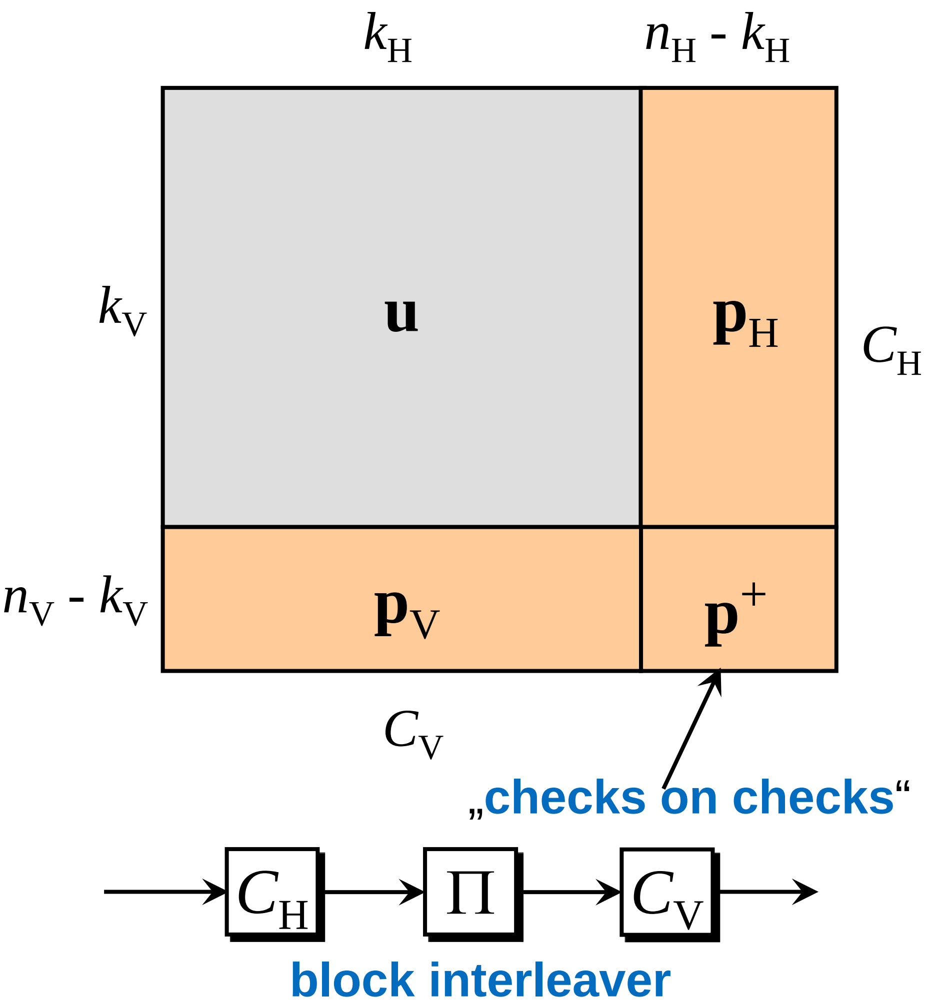
original concatenation:

$d_{min} = 3$

optimized concatenation:

$d_{min} = 4$

Serial Code Concatenation: Product Codes



- Information bits arranged in $(k_\mathrm{V},k_\mathrm{H})$-matrix $\mathbf{u}$
- Row-wise encoding with systematic $(n_\mathrm{H}, k_\mathrm{H}, d_\mathrm{H})$-code C_H of rate $k_\mathrm{H}/ n_\mathrm{H}$
  → each row contains a code word
- Column-wise encoding with systematic $(n_\mathrm{V}, k_\mathrm{V}, d_\mathrm{V})$-code C_V of rate $k_\mathrm{V} / n_\mathrm{V}$
  → each column contains a code word
- Entire code rate:

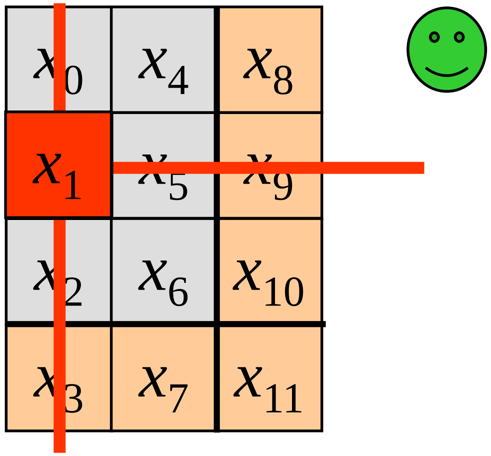
$$\boxed{R_c = \frac{k_\mathrm{H} \cdot k_\mathrm{V}}{n_\mathrm{H} \cdot n_\mathrm{V}} = R_{c,\mathrm{H}} \cdot R_{c,\mathrm{V}}}$$

- Minimum Hamming distance:

$$\boxed{d_{\min} = d_{\min,\mathrm{H}} \cdot d_{\min,\mathrm{V}}}$$

Serial Code Concatenation: Examples of Product Codes

(12,6,4) product code

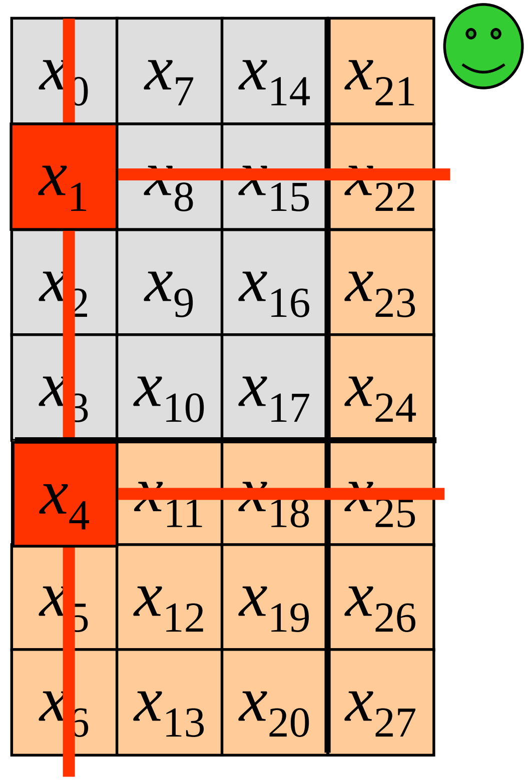


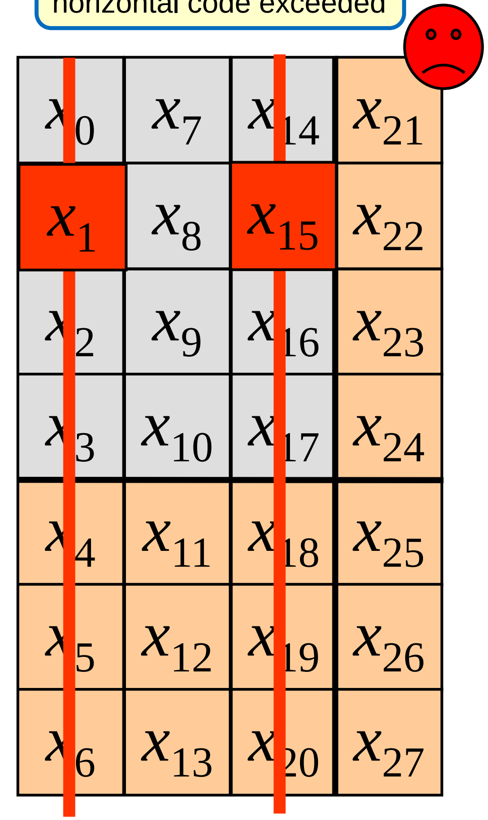
- Horizontal: (3,2,2)-SPC code
- Vertical: (4,3,2)-SPC code
- Code rate: 1/2
- $d_{\text{min}} = 2 \cdot 2 = 4$
- Correction of 1 error & detection of 3 errors possible

Interleaver combines 3 info words → increase of eff. block length

(28,12,6) product code

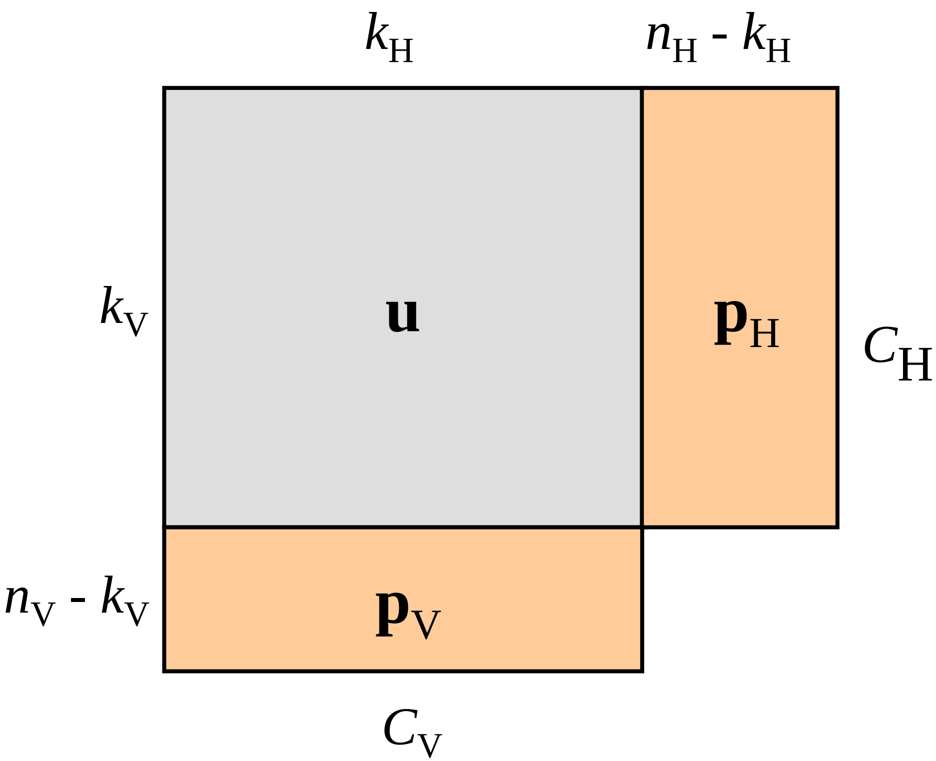
Detection capability of horizontal code exceeded

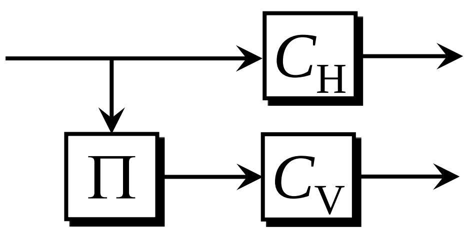




- Horizontal: (4,3,2)-SPC code
- Vertical: (7,4,3)-Hamming code
- $d_{\text{min}} = 2 \cdot 3 = 6$ ➔ correction of 2 errors possible

Parallel Code Concatenation: Modified Product Codes





- Information bits $\mathbf{u}$ row-wise and column-wise encoded with C_H and C_V, respectively
- Parity check bits of component codes not encoded twice **(no checks on checks)**
- Entire code rate

$$R_c = \frac{k_\mathrm{H} \cdot k_\mathrm{V}}{n_\mathrm{H} \cdot n_\mathrm{V} - (n_\mathrm{H} - k_\mathrm{H}) \cdot (n_\mathrm{V} - k_\mathrm{V})} = \frac{1}{1/R_{c,\mathrm{H}} + 1/R_{c,\mathrm{V}} - 1}$$

- Minimum Hamming distance:

$$d_{\min} = d_{\min,\mathrm{H}} + d_{\min,\mathrm{V}} - 1$$

Parallel Code Concatenation: Examples

modified (11,6,3) product code

x_0	x_4	x_8
x_1	x_5	x_9
x_2	x_6	x_{10}
x_3	x_7	

$d_{\min} = 3$

- Horizontal: (3,2,2) SPC code
- Vertical: (4,3,2) SPC code
- Code rate: 6/11
- $d_{\min} = 2 + 2 - 1 = 3$
- 1 error correctable

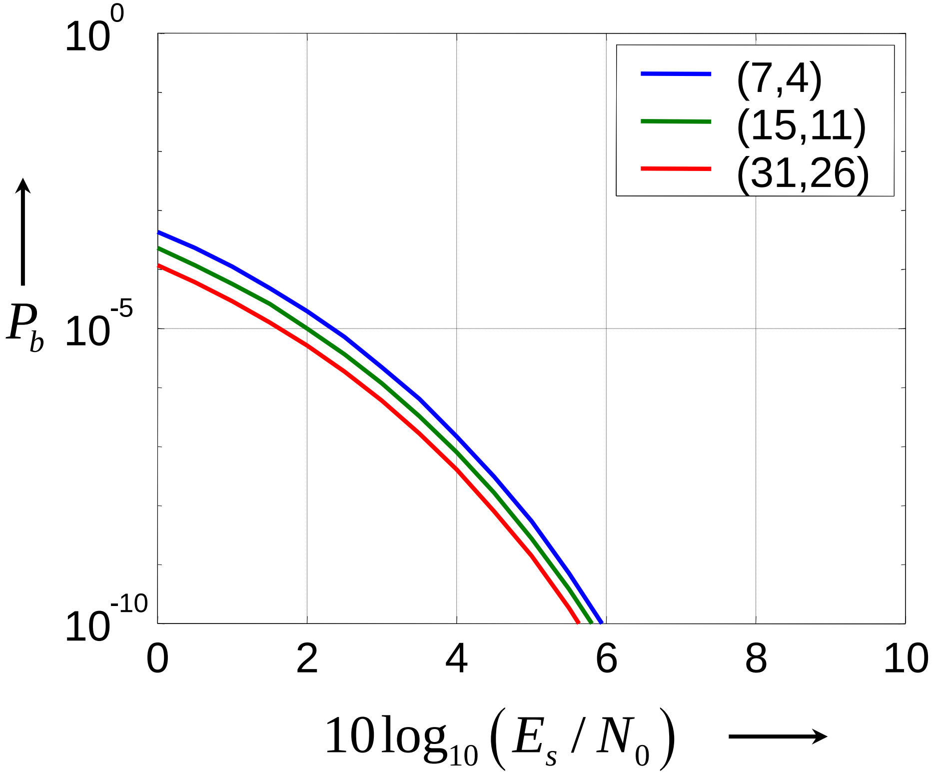
modified (25,12,4) product code

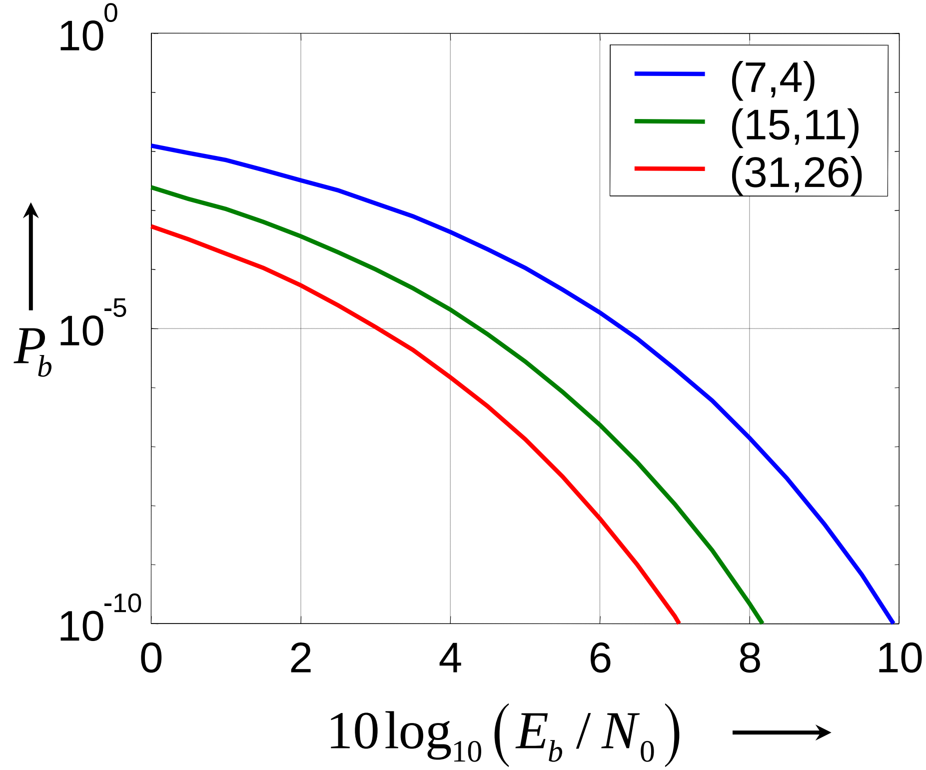
x_0	x_7	x_{14}	x_{21}
x_1	x_8	x_{15}	x_{22}
x_2	x_9	x_{16}	x_{23}
x_3	x_{10}	x_{17}	x_{24}
x_4	x_{11}	x_{18}	
x_5	x_{12}	x_{19}	
x_6	x_{13}	x_{20}	

- Horizontal: (4,3,2) SPC code
- Vertical: (7,4,3) Hamming code
- $d_{\min} = 2 + 3 - 1 = 4 \rightarrow$ 1 error correctable

Union Bound on Bit Error Rate for Product Codes

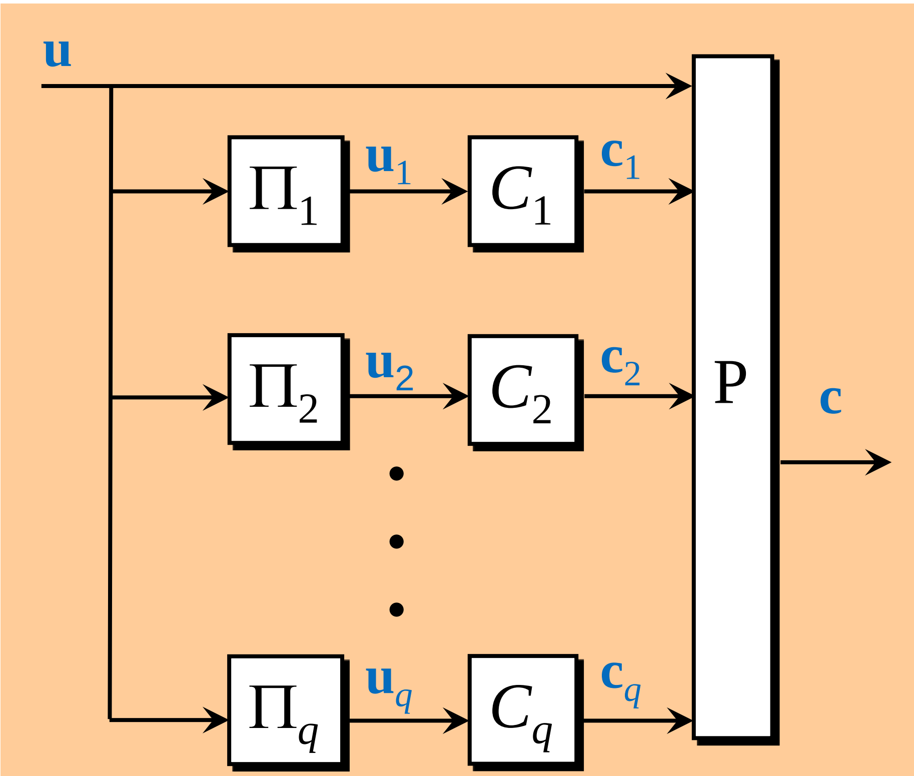
- Product codes using same $(n,k,3)$-Hamming code
- Only taking into account minimum distance $d_{\min}=3+3-1=5$
  → results only valid for high signal to noise ratios





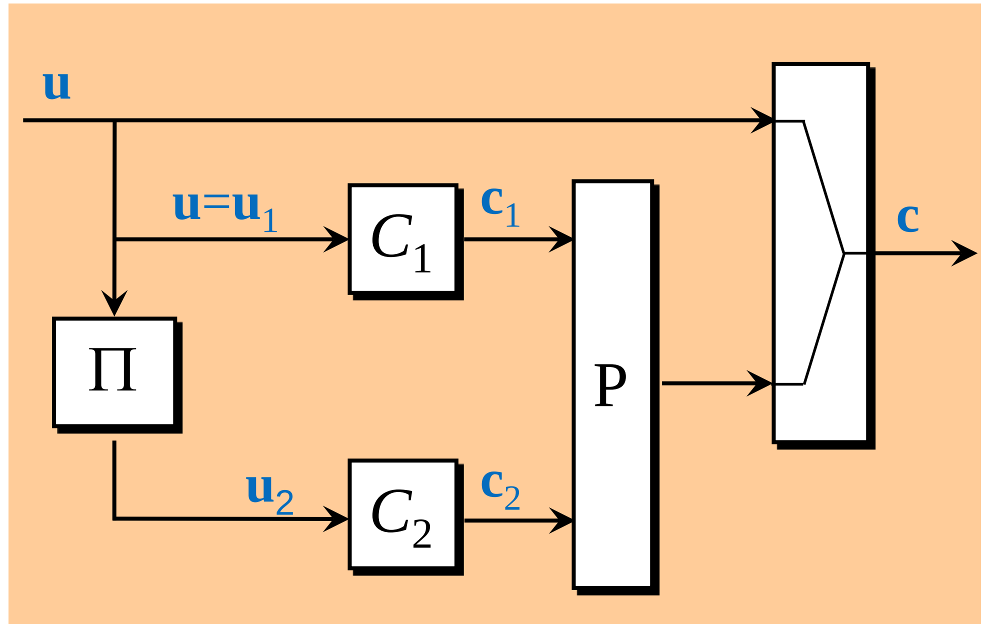
Parallel Code Concatenation: Turbo Codes

General structure with q constituent codes



- Presented in 1993 by Berrou, Glavieaux, Thitimajshima

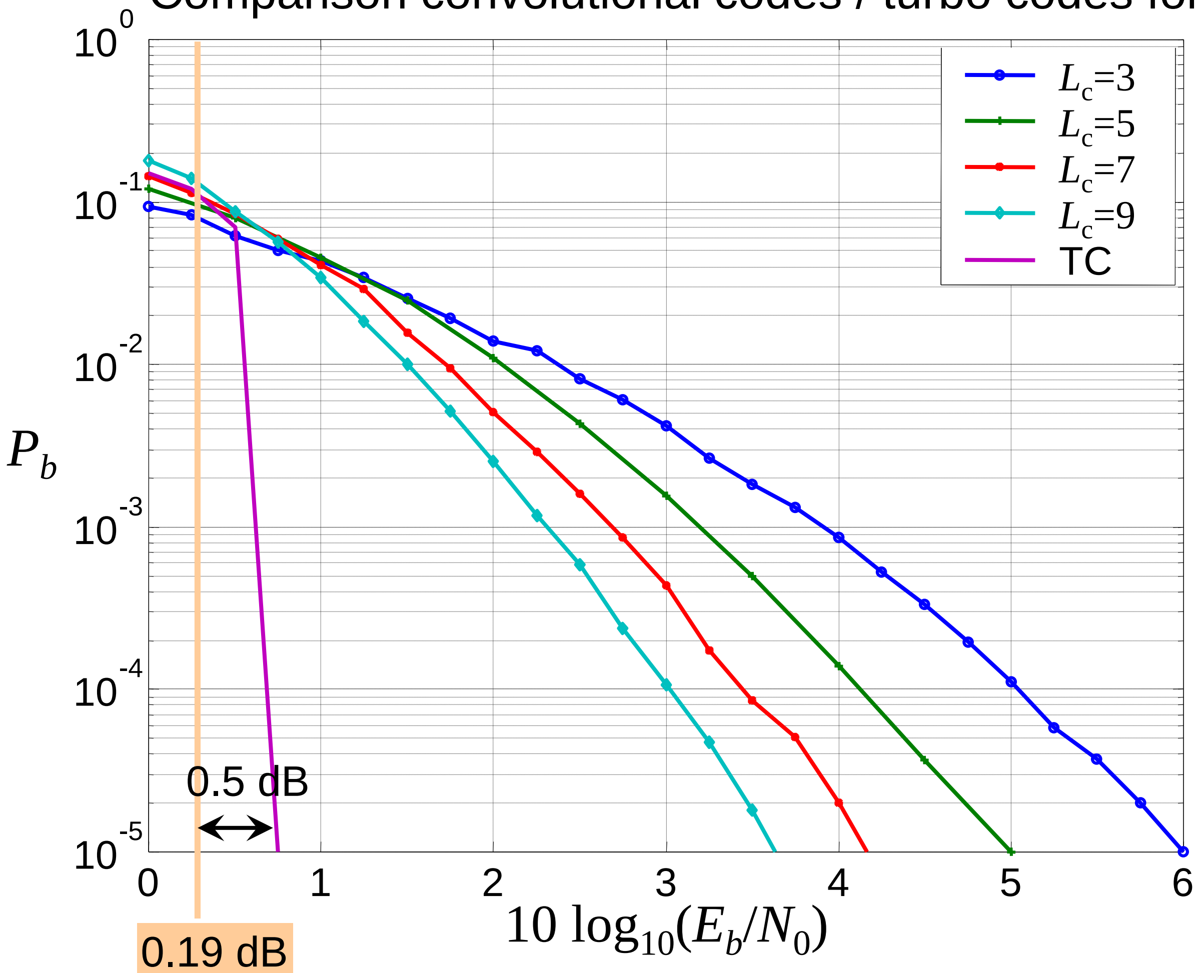
special case with 2 constituent codes



- Interleaver Π_1 neglectable
- Information bits generally not punctured
- Code rate: $R_c = \dfrac{1}{1/R_{c,1} + 1/R_{c,2} - 1}$

Potential of Turbo Codes

Comparison convolutional codes / turbo codes for R_c=1/2



- Optimized interleaver of length 256 x 256 = 65536 bits
- For this interleaver, gain of nearly 3 dB over convolutional code with $L_c = 9$
- Gap to Shannon‘s channel capacity only 0.5 dB ($C = 0.5$ at $E_b/N_0 = 0.19$ dB)
- Tremendous performance loss for smaller interleavers
- World record: 0.08 dB gap to Shannon capacity by Stephan ten Brink

Influence of Constituent Codes

- Systematic recursive convolutional encoders employed in turbo codes
  - Constituent codes generate only parity bits
  - Conventionally codes with small constraint length ($3 \le L_c \le 5$) and rate $R_c = \frac{1}{n}$ (codes of larger rate can be achieved by puncturing)
  - Error probability depends on interleaver size L_π and minimum input weight w_{min} of constituent encoders that leads to finite output weight

$$\boxed{P_b \sim L_\pi^{1-w_{\min}}}$$

  - Only recursive encoders require at least $w_{\min} = 2$ for finite output weight
  - Interleaving gain only achievable for recursive encoders due to $P_b \sim L_\pi^{-1}$
  - Nonrecursive encoders with $w_{\min} = 1$ do not gain from enlarging interleaver size ($P_b \sim L_\pi^0$)

RSC-Encoders are used as constituent codes
→ performance improves with length of interleaver!

Influence of Constituent Codes

- Instead of free distance d_f the **effective distance** d_{eff} is crucial

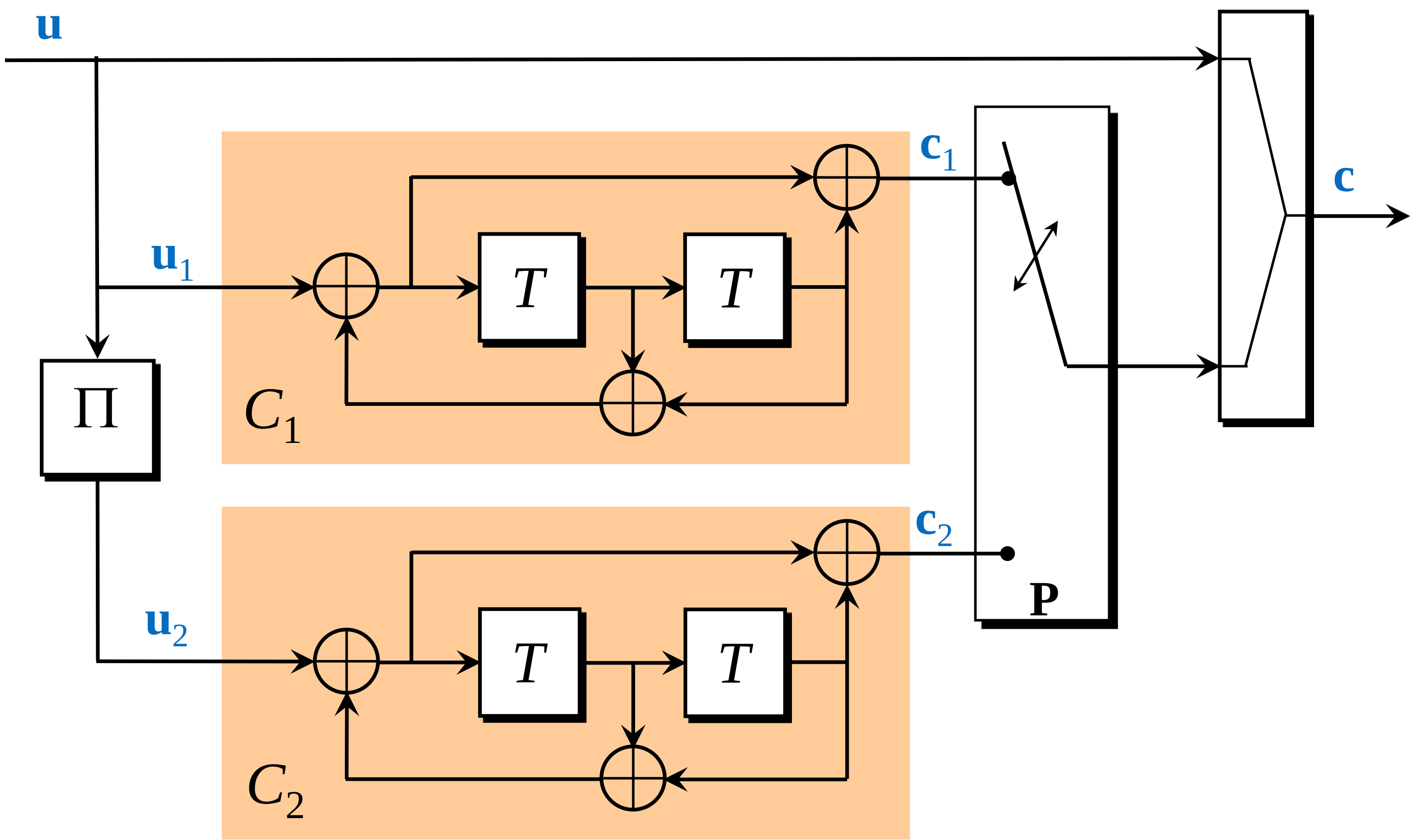
$$d_{\text{eff}} = w_{\min} + 2 \cdot c_{\min}$$

- Interpretation: Turbo codes are systematic codes
  - Total weight of code words depends on weight of information bits $w_{\min}$
  - $c_{\min}$ denotes minimum weight of parity bits of one encoder for input weight $w_{\min} = 2$
  - Assuming same constituent codes, minimum weight for $w_{\min} = 2$ is given by d_{eff}

- Consequence:
  - Suitable constituent codes should maximize parity weight for input weight $w_{\min} = 2$
  - Aim is achieved if feedback polynomial of constituent encoders is prime
    - Shift register generates sequence of maximum length (m-sequence)
      → may have larger weight than shorter sequences

Feedback polynomial of constituent encoders should be prime!

Example of Turbo Code with 2 Codes ($L_c = 3$), $R_c = 1/2$



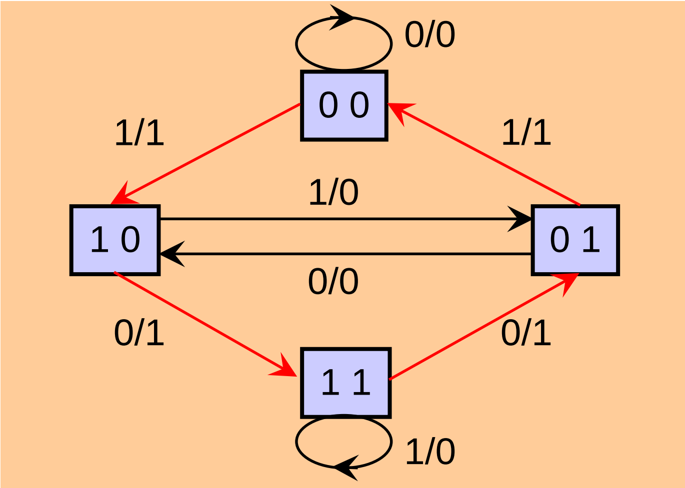
$$g_1 = 5_8$$

$$\underline{g}_2 = 7_8$$

$$\mathbf{P} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

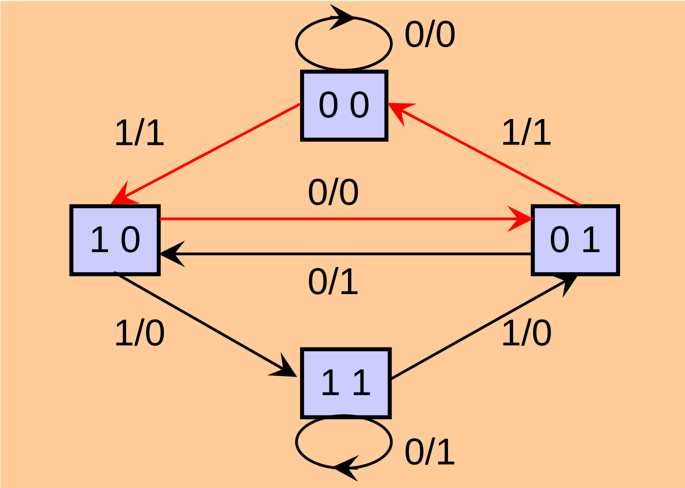
Example of Turbo Code with 2 Codes ($L_c = 3$), $R_c = 1/2$

- Recursive polynomial: $g_2(D) = 1 + D + D^2$
  - $g_2(D)$ is prime
    $g_2(0) = 1 + 0 + 0 = 1$ and $g_2(1) = 1 + 1 + 1 = 1$
  - Shift register achieves sequence of maximum length (m-sequence) with $L = 2^2\text{-}1=3$
  - Max dist. $d_{\text{eff}}^{\max} = w_{\min} + 2 \cdot (L+1) = 2 + 2 \cdot 4 = 10$
  - $\mathbf{u} = [1\ 0\ 0\ 1] \rightarrow \mathbf{c}_1 = [1\ 1\ 1\ 1]$



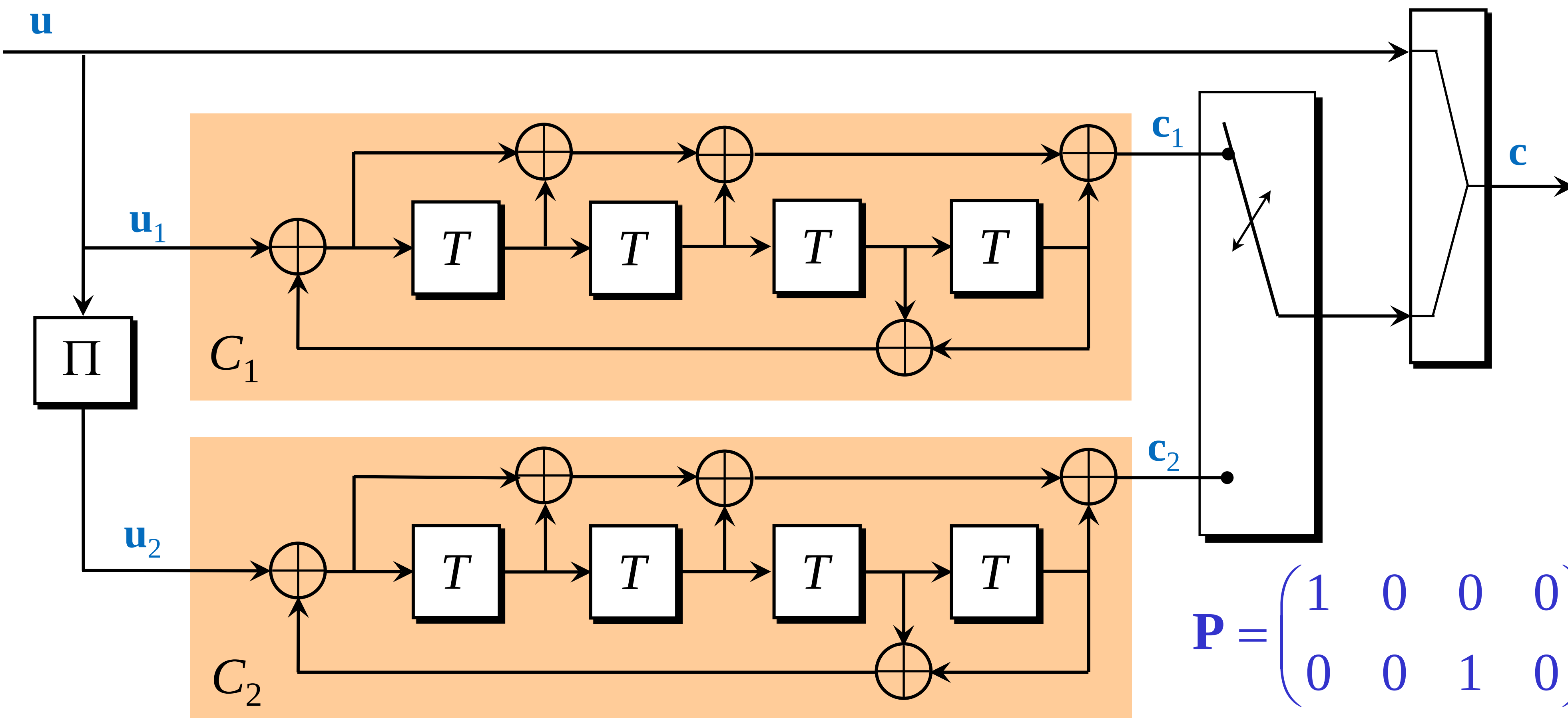
- Recursive polynomial: $g_1(D) = 1 + D^2$
  - $g_1(D) = (1+D)(1+D) \rightarrow$ non-prime
  - Shift register generates sequence of length $L = 2$
  - Max dist. $d_{\text{eff}}^{\max} = w_{\min} + 2 \cdot (L+1) = 2 + 2 \cdot 3 = 8$
  - $\mathbf{u} = [1\ 0\ 1] \rightarrow \mathbf{c}_1 = [1\ 0\ 1]$

Feedback polynomial $g_1(D)$ would lead to degraded performance!



Example of Turbo Code with 2 Codes ($L_c = 5$), $R_c = 2/3$

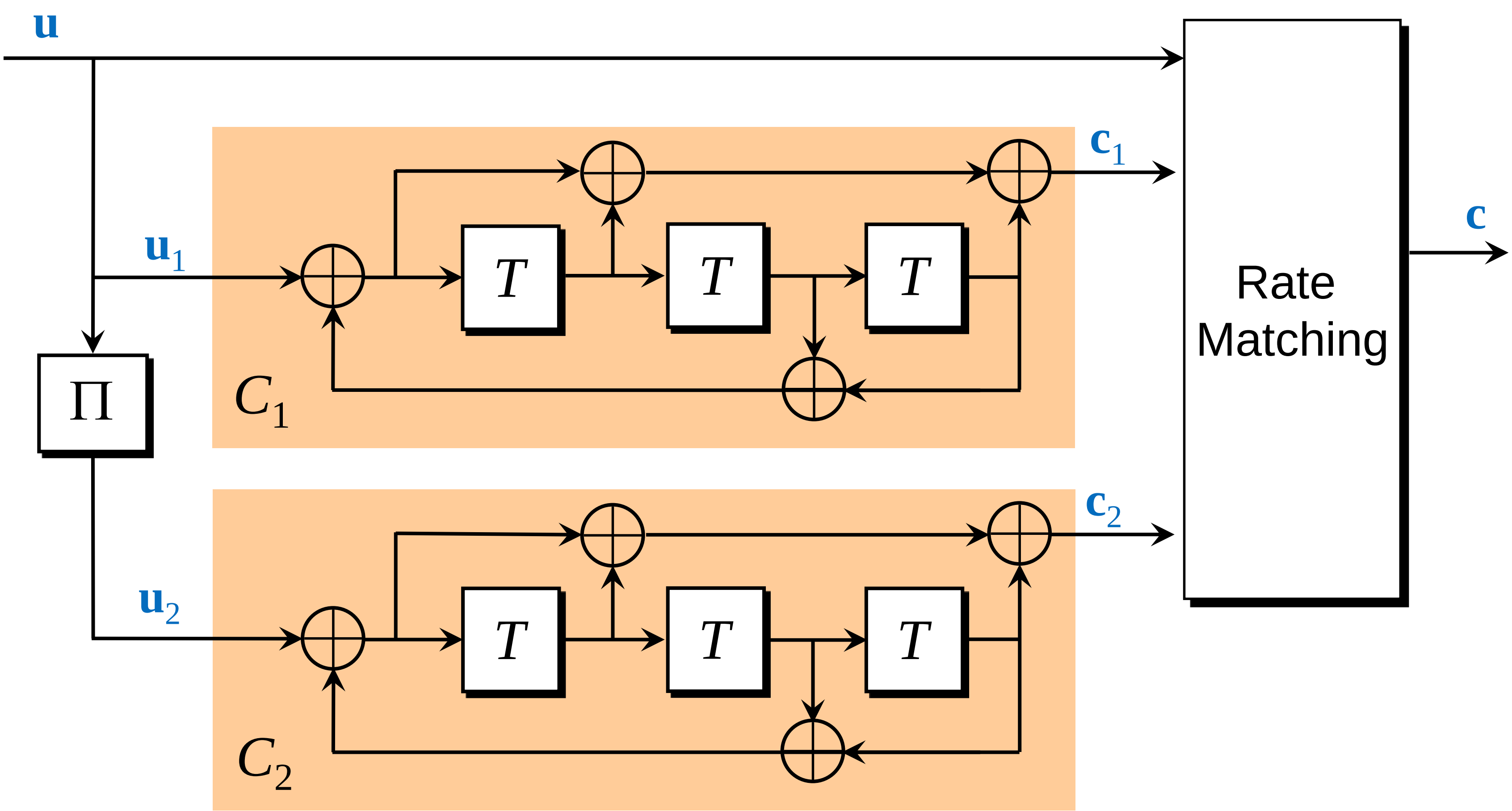
$\underline{g_1} = 23_8 \qquad g_2 = 35_8$



$$\mathbf{P} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

LTE Turbo Code with 2 Codes ($L_c = 4$)

$$\underline{g}_1 = 1 + D^2 + D^3 = 13_8 \qquad g_2 = 1 + D + D^3 = 15_8$$



Influence of Interleaver

$$P_b \le \frac{1}{2}\sum_d c_d \cdot \operatorname{erfc}\left(\sqrt{d \cdot R_c \frac{E_b}{N_0}}\right)$$

c_d: total number of nonzero info bits associated with code sequences of Hamming weight d

- Avoiding output sequences with low Hamming weight at both encoders
  - If output $\mathbf{c}_1$ of C_1 has low Hamming weight → permutation of input sequence $\mathbf{u}_2$ for C_2 should result in output sequence $\mathbf{c}_2$ with high Hamming weight
  - Higher total average Hamming weight / Hamming distance d
- Interleaver directly influences minimum distance
- Number of sequences with low weight reduced due to interleaving
  - Small coefficients c_d
  - Even more important than minimum distance that acts only asymptotically
- Randomness of interleaver is important
  - Simple block interleavers perform bad due to symmetry
  - Pseudo-random interleavers are much better → random codes (→ Shannon)

Distance Properties of Turbo Codes: Definitions

- General IOWEF (*Input Output Weight Enumerating Function*) of encoder:

$$A(W,D)=\sum_{w=0}^{k}\sum_{d=0}^{n}A_{w,d}\cdot W^{w}\cdot D^{d}$$

$A_{w,d}$: number of code words with input weight w and output weight d

- Conditioned IOWEF's (specific input weight w or specific output weight d):

$$A(w,D)=\sum_{d=0}^{n}A_{w,d}\cdot D^{d}$$

$$A(W,d)=\sum_{w=0}^{k}A_{w,d}\cdot W^{w}$$

- Important for parallel concatenation: weight c of parity bits

$$A(W,C)=\sum_{w}\sum_{c}A_{w,c}\cdot W^{w}\cdot C^{c}$$

with $d=w+c$

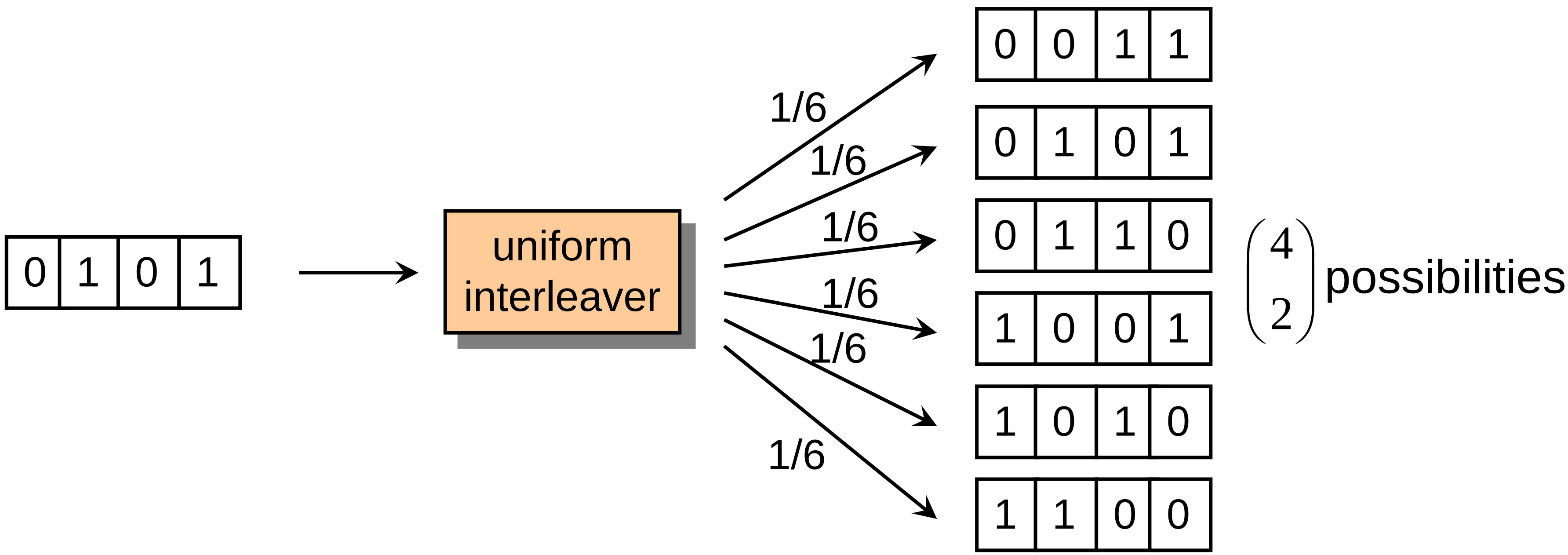
All encoders have same input weight w
Encoders generate only parity bits
→ consider weight c of parity bits

  - Corresponding conditioned IOWEF:

$$A(w,C)=\sum_{c}A_{w,c}\cdot C^{c}$$

Distance Properties of Turbo Codes: Uniform Interleaver

- Problem: concrete interleaver has to be considered for distance spectrum / IOWEF
  $\rightarrow$ determination of IOWEF computationally expensive
- **Uniform interleaver (UI)**: theoretic device comprising all possible permutations



- UI provides average distance spectrum (incl. good and bad interleavers)

Distance Properties of Turbo Codes: Results

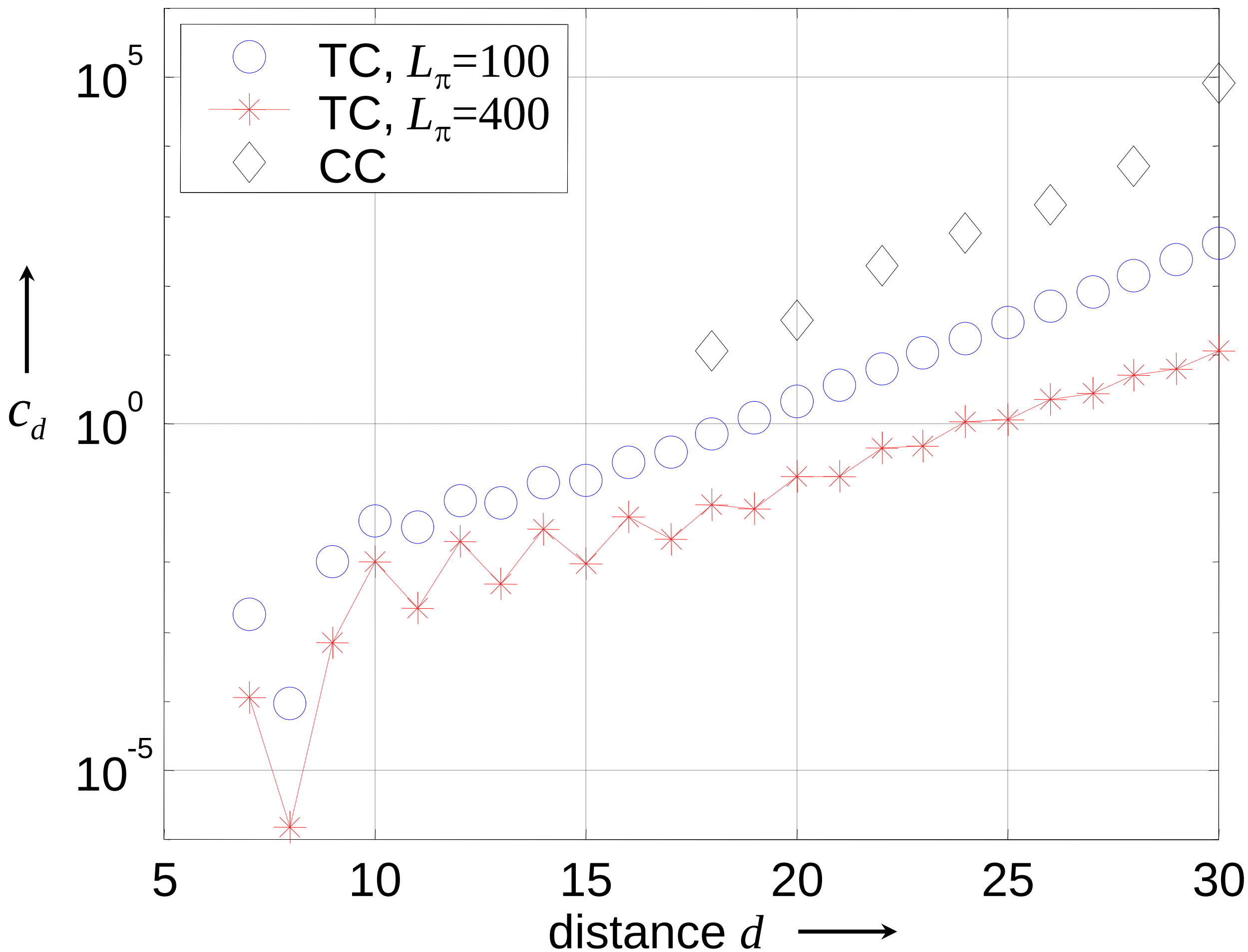
- Parallel concatenation:
  - Both encoders have same input weight w
  - Weights c_1 and c_2 of encoder outputs are added
    - $A_1(w,C)\cdot A_2(w,C)$ combines output sequences with same input weight w and covers all possible combinations of output sequences (uniform interleaver)
    - Denominator achieves averaging w.r.t. number of permutations of w ones in length L_π

$$A^{\text{par}}(w,C) = \frac{A_1(w,C)\cdot A_2(w,C)}{\binom{L_\pi}{w}} = \sum_c A^{\text{par}}_{w,c}\cdot C^c \quad \Rightarrow \quad c_d = \sum_{w+c=d} \frac{w}{L_\pi}\cdot A^{\text{par}}_{w,c}$$

- Serial concatenation:
  - Output weight ℓ of outer encoder equals input weight of inner encoder

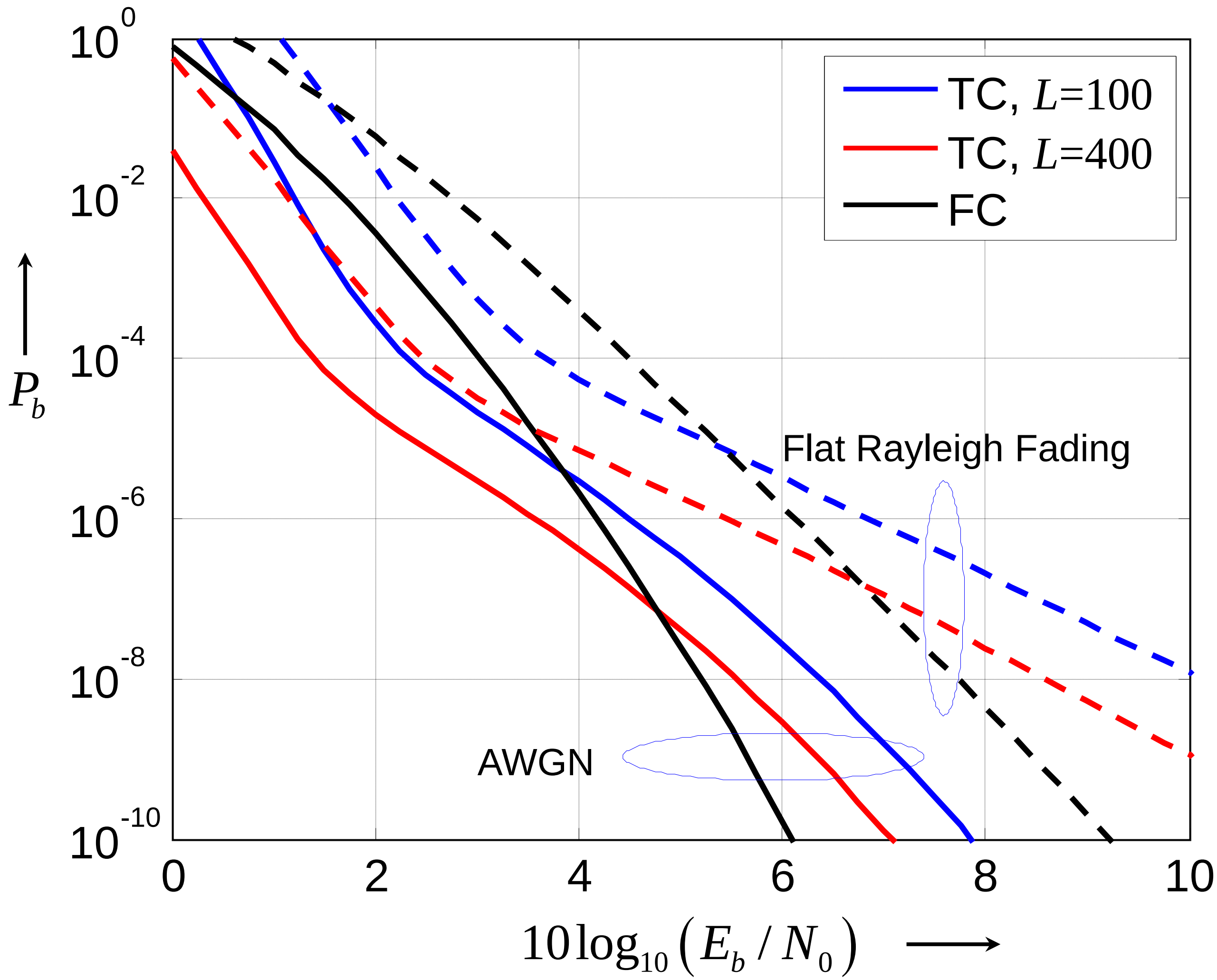
$$A^{\text{ser}}(W,D) = \sum_\ell \frac{A_1(W,\ell)\cdot A_2(\ell,D)}{\binom{L_\pi}{\ell}} = \sum_w \sum_d A^{\text{ser}}_{w,d}\cdot W^w \cdot D^d \quad \Rightarrow \quad c_d = \sum_w \frac{w}{L_\pi \cdot R_c^1}\cdot A^{\text{ser}}_{w,d}$$

Distance Properties of Turbo Codes



- Codes
  - Turbo Code
    $\mathbf{g}_1 = 5_8$, $\mathbf{g}_2 = 7_8$
  - Convolutional Code with L_c=9
  - R_c=1/3
- Observations
  - UI → $c_d < 1$ is possible
  - TC has lower d_f but coefficients c_d are much smaller
    → effect becomes more obvious with increasing interleaver length L_π

Analytical Error Rate Estimation of Turbo Codes



- Observations
  - For small SNR the TC outperforms CC significantly
  - Gain increases with L_π
  - For increasing SNR the BER of TC flattens, whereas the curve of CC decreases
- Explanations
  - d_f dominates BER for large SNR
  - For small SNR the number of sequences with specific weight is of larger importance

Decoding of Concatenated Codes

- Definition of Soft-Information
- L-Algebra
- General Approach for Soft-Output Decoding
- Soft-Output Decoding using the Dual Code
- Soft-Output Decoding for (4,3,2)-SPC-Code
- BCJR Algorithm for Convolutional Codes

Decoding of Concatenated Codes

- Optimum Maximum Likelihood Decoding of concatenated codes is too complex
- Constituent codes C_1 and C_2 are decoded by separated decoders D_1 and D_2
- Decoders D_1 and D_2 are allowed to exchange "information" in to improve their performance
  → probability of information and/or code bits is of interest
  → **soft output decoding** is required!
- What is a useful soft output?
  - Assumption: uncoded transmission over AWGN channel $y = x + n$
  - BPSK modulation

$$x = 1 - 2u \quad \Rightarrow \quad \begin{array}{lcl} u = 0 & \rightarrow & x = +1 \\ u = 1 & \rightarrow & x = -1 \end{array}$$

$\oplus$	0	1
0	0	1
1	1	0

⇔

$\cdot$	+1	-1
+1	+1	-1
-1	-1	+1

  - **MAP criterion** (Maximum a posteriori) considers unequal distribution of symbols

$$\Pr\{u = 0 \mid y\} \gtrless \Pr\{u = 1 \mid y\} \quad \Leftrightarrow \quad \Pr\{x = +1 \mid y\} \gtrless \Pr\{x = -1 \mid y\}$$

Decoding of Concatenated Codes

- Conditional Probability $\Pr\left\{x=+1\middle|y\right\} = p\left\{x=+1,y\right\}/\Pr\left\{y\right\}$

$$\frac{p\{x=+1,y\}}{\Pr\{y\}} \gtrless \frac{p\{x=-1,y\}}{\Pr\{y\}} \quad \Rightarrow \quad \frac{p\{x=+1,y\}}{p\{x=-1,y\}} = \frac{p\{y\,|\,x=+1\}}{p\{y\,|\,x=-1\}} \cdot \frac{\Pr\{x=+1\}}{\Pr\{x=-1\}} \gtrless 1$$

- **Log-Likelihood-Ratio** (**LLR**) (or L-values) derived by **Hagenauer**

$$L(\hat{x}) = L(x,y) = L(x\,|\,y) = \ln\frac{p\{x=+1,y\}}{p\{x=-1,y\}} \gtrless 0$$

$$= \underbrace{\ln\frac{p\{y\,|\,x=+1\}}{p\{y\,|\,x=-1\}}}_{L(y|x)} + \underbrace{\ln\frac{\Pr\{x=+1\}}{\Pr\{x=-1\}}}_{L_a(x)} = L(y\,|\,x) + L_a(x)$$

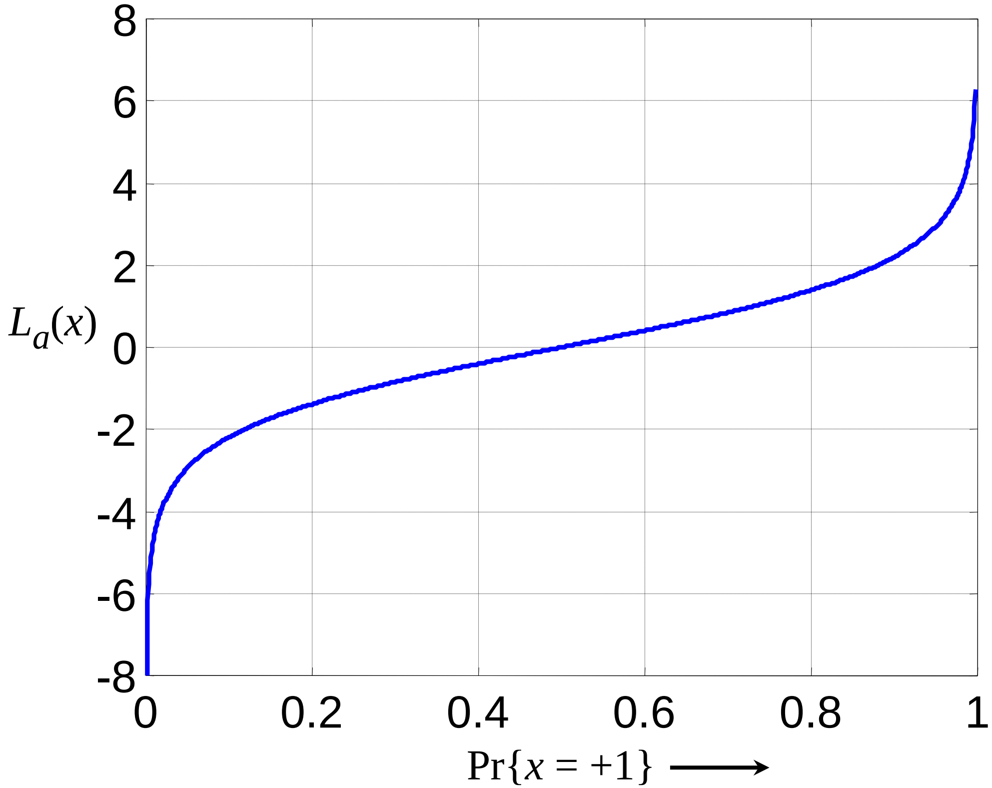
Joachim Hagenauer

  - Sign $\operatorname{sgn}\{L(\hat{x})\}$ corresponds to hard decision
  - Magnitude $|L(\hat{x})|$ indicates reliability of hard decision
  - Another possible definition would be (not used)
    $L(x) = \Pr\{x=+1\} - \Pr\{x=-1\}$

Addition of LLRs requires <u>statistically independency</u> of variables!

Log-Likelihood-Ratio

- For an uncoded transmission the LLR consists of two components
  - $L(y|x)$ depends on **channel statistics** and therefore on the received signal y
  - $L_a(x)$ represents **a-priori knowledge** about symbol x



$$L_a(x) = \ln \frac{\Pr\{x=+1\}}{\Pr\{x=-1\}}$$

- Symmetric with respect to (0,5 ; 0)
- $\Pr\{x=+1\} > 0{,}5$
  → +1 more likely than -1
  → positive $L_a(x)$
- The larger the difference between $\Pr\{x=+1\}$ and $\Pr\{x=-1\}$ the larger $L_a(x)$
  → suitable value for reliability
- $\Pr\{x=+1\} = 0{,}5$ → $L_a(x) = 0$ → decision would be random

LLR for a Memoryless Channel

- Memoryless channel (AWGN or 1-path fading channel) $y = \alpha\, x + n$
- Channel information

$$L(y|x) = \ln \frac{p\{y \mid x=+1\}}{p\{y \mid x=-1\}} = \ln \frac{\exp\left(-\frac{1}{2\sigma^2}\left(y-\alpha\sqrt{E_s/T_s}\right)^2\right)}{\exp\left(-\frac{1}{2\sigma^2}\left(y+\alpha\sqrt{E_s/T_s}\right)^2\right)}$$

$$= \frac{1}{2\sigma^2}\left(y+\alpha\sqrt{E_s/T_s}\right)^2 - \frac{1}{2\sigma^2}\left(y-\alpha\sqrt{E_s/T_s}\right)^2$$

$$= \frac{4\alpha\, y\sqrt{E_s/T_s}}{2\sigma^2} = 4\alpha\, y \frac{\sqrt{E_s/T_s}}{N_0/T_s} = \underbrace{4|\alpha|^2 \frac{E_s}{N_0}}_{L_{ch}} y'$$

with $\sigma^2 = \frac{N_0}{2T_s}$

normalized received signal

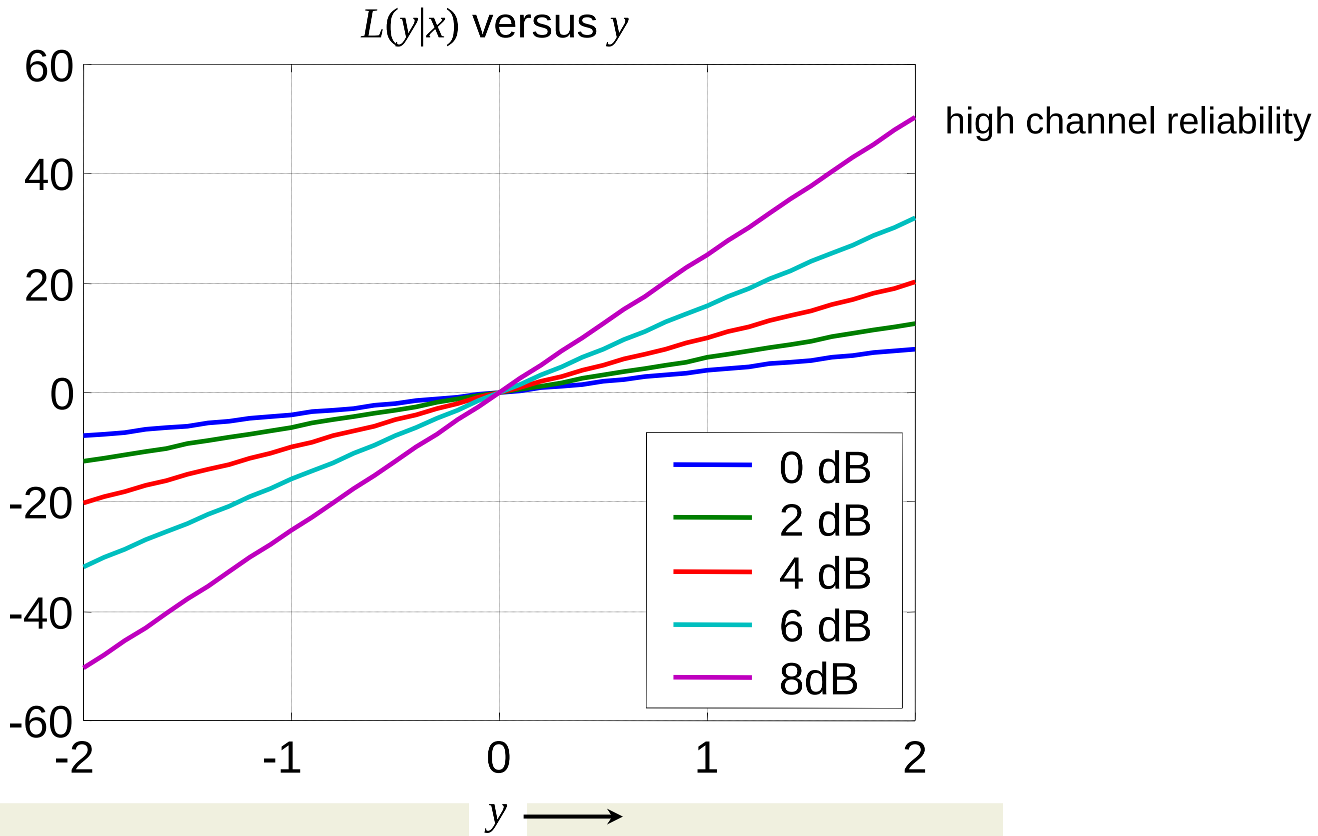
$$y' = \frac{y}{|\alpha|\sqrt{E_s/T_s}}$$

- L_{ch}= reliability of the channel (depends on SNR E_S/N_0 and channel gain $|\alpha|^2$)

LLR for a Memoryless Channel

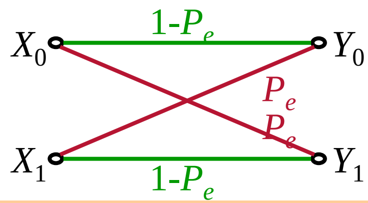
- **Reliability of channel**: $L_{ch} = 4|\alpha|^2 \frac{E_s}{N_0}$
- LLR is simply a scaled version of the matched filter → motivation for $\ln$

$$L(y|x) = 4|\alpha|^2 \frac{E_s}{N_0} y'$$

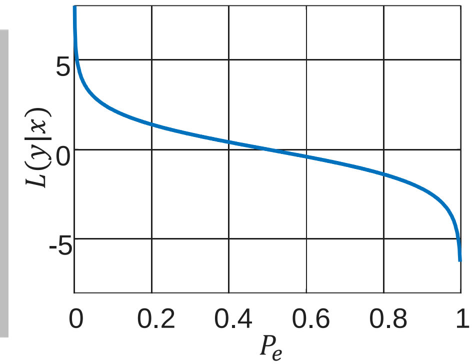


LLRs for BSC and BEC

- Binary Symmetric Channel (BSC)

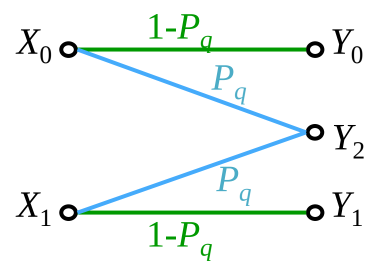


$$L\left(y|x\right) = \ln\frac{p\{y\,|\,x=+1\}}{p\{y\,|\,x=-1\}} = \begin{cases} \ln\dfrac{1-P_e}{P_e} & \text{for } y=Y_0=+1 \\ \ln\dfrac{P_e}{1-P_e} & \text{for } y=Y_1=-1 \end{cases} = y\cdot\ln\frac{1-P_e}{P_e}$$



- Binary Erasure Channel (BEC)

$$L\left(y|x\right) = \begin{cases} \ln\dfrac{1-P_q}{0} & \text{for } y=Y_0 \\ \ln\dfrac{P_q}{P_q} & \text{for } y=Y_2 \\ \ln\dfrac{0}{1-P_q} & \text{for } y=Y_1 \end{cases} = \begin{cases} +\infty & \text{for } y=Y_0 \\ 0 & \text{for } y=Y_2 \\ -\infty & \text{for } y=Y_1 \end{cases}$$



Relation between LLRs and Probabilities (1)

- Matched filter corresponds to LLR → Task: Find arithmetic to perform operation with respect to LLR instead of probabilities → L-algebra by Hagenauer
- Basic relation
  - Using completeness ($\Pr\{x = +1\} + \Pr\{x = -1\} = 1$) in LLR

$$L(\hat{x}) = L(x \mid y) = \ln \frac{\Pr\{x=+1 \mid y\}}{\Pr\{x=-1 \mid y\}} = \ln \frac{\Pr\{x=+1 \mid y\}}{1-\Pr\{x=+1 \mid y\}} = \ln \frac{1-\Pr\{x=-1 \mid y\}}{\Pr\{x=-1 \mid y\}}$$

$$\Rightarrow \boxed{\Pr\{x=+1 \mid y\} = \frac{e^{L(\hat{x})}}{1+e^{L(\hat{x})}} = \frac{1}{1+e^{-L(\hat{x})}}}$$

$$\Rightarrow \boxed{\Pr\{x=-1 \mid y\} = \frac{1}{1+e^{L(\hat{x})}}}$$

- With respect to symbol $x \in \{+1,-1\}$ the general relation holds

$$\boxed{\Pr\{x=i \mid y\} = \frac{e^{L(\hat{x})/2}}{1+e^{L(\hat{x})}} \cdot e^{i \cdot L(\hat{x})/2} = \frac{1}{1+e^{-\mathrm{sgn}(i) \cdot L(\hat{x})}} \quad \text{with } i \in \{-1,+1\}}$$

Relation between LLRs and Probabilities (2)

- Probability of a correct decision
  - For $x = +1$ decision is correct, if $L(\hat{x})$ is positive

$$\Pr\left\{\hat{x} \text{ correct} \,\middle|\, x = +1\right\} = \frac{e^{L(\hat{x})}}{1+e^{L(\hat{x})}} = \frac{e^{|L(\hat{x})|}}{1+e^{|L(\hat{x})|}}$$

  - For $x = -1$ decision is correct, if $L(\hat{x})$ is negative

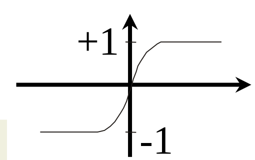
$$\Pr\left\{\hat{x} \text{ correct} \,\middle|\, x = -1\right\} = \frac{1}{1+e^{L(\hat{x})}} = \frac{1}{1+e^{-|L(\hat{x})|}} = \frac{e^{|L(\hat{x})|}}{1+e^{|L(\hat{x})|}}$$

$$\Rightarrow \quad \boxed{\Pr\left\{\hat{x} \text{ is correct}\right\} = \frac{e^{|L(\hat{x})|}}{1+e^{|L(\hat{x})|}}}$$

- **Soft bit**: expected value for antipodal tx signal

$$\boxed{\lambda = \mathrm{E}\left\{\hat{x}\right\} = \sum_{i=\pm 1} i \cdot \Pr\left\{\hat{x} = i\right\} = (+1)\frac{e^{L(\hat{x})}}{1+e^{L(\hat{x})}} + (-1)\frac{1}{1+e^{L(\hat{x})}} = \frac{e^{L(\hat{x})}-1}{e^{L(\hat{x})}+1} = \tanh\frac{L(\hat{x})}{2}}$$

$$\Rightarrow \quad \boxed{\Pr\left\{\hat{x} = +1\right\} = \frac{\lambda+1}{2}}$$



L-Algebra

- Parity bits are generated by modulo-2-sums of certain information bits
  → how can we calculate the L-value of a parity bit? → Hagenauer
- Assumption: Single parity check code (SPC) $p = u_1 \oplus u_2$ ➡ $L(p) = ?$
  - x_1 and x_2 are statistically independent

$$2\,\mathrm{artanh}(x) = \ln \frac{1+x}{1-x}$$

$$\lambda = \tanh\left(\tfrac{x}{2}\right) = \frac{e^x - 1}{e^x + 1}$$

$$L(p) = L(u_1 \oplus u_2) = \ln \frac{\Pr\{u_1 \oplus u_2 = 0\}}{\Pr\{u_1 \oplus u_2 = 1\}} = \ln \frac{\Pr\{x_1 \cdot x_2 = +1\}}{\Pr\{x_1 \cdot x_2 = -1\}} = L(x_1 \cdot x_2)$$

$$L(x_1 \cdot x_2) = \ln \frac{\Pr\{x_1 = +1\} \cdot \Pr\{x_2 = +1\} + \Pr\{x_1 = -1\} \cdot \Pr\{x_2 = -1\}}{\Pr\{x_1 = +1\} \cdot \Pr\{x_2 = -1\} + \Pr\{x_1 = -1\} \cdot \Pr\{x_2 = +1\}} = \ln \frac{\frac{\Pr\{x_1 = +1\}}{\Pr\{x_1 = -1\}} \cdot \frac{\Pr\{x_2 = +1\}}{\Pr\{x_2 = -1\}} + 1}{\frac{\Pr\{x_1 = +1\}}{\Pr\{x_1 = -1\}} + \frac{\Pr\{x_2 = +1\}}{\Pr\{x_2 = -1\}}}$$

$$L(x_1 \cdot x_2) = \ln \frac{e^{L(x_1)} \cdot e^{L(x_2)} + 1}{e^{L(x_1)} + e^{L(x_2)}} = \ln \frac{e^{L(x_1)+L(x_2)} + 1}{e^{L(x_1)} + e^{L(x_2)}}$$

boxplus operation

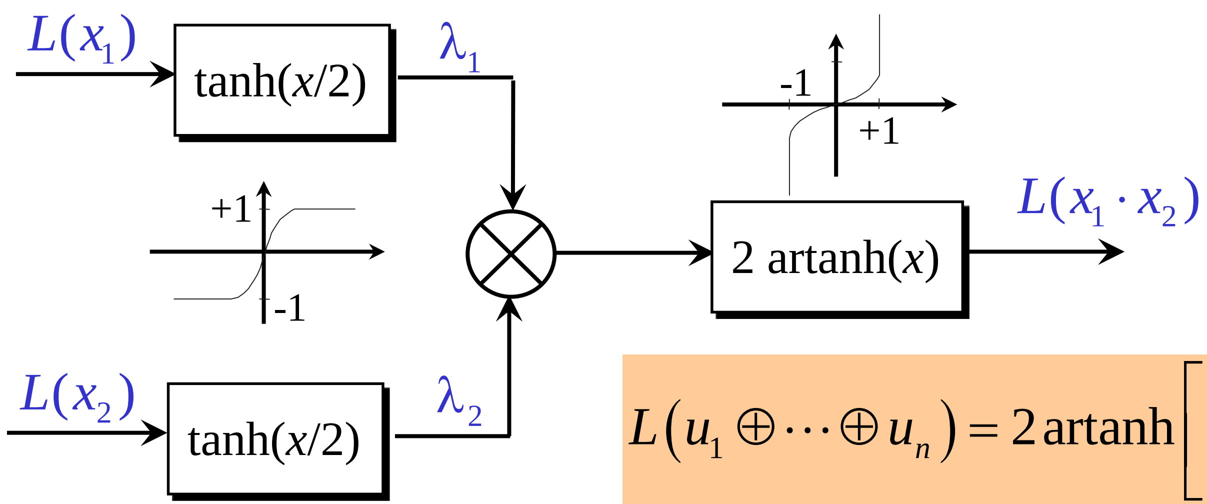
$$= 2\,\mathrm{artanh}\left[\tanh\left(\frac{L(x_1)}{2}\right) \cdot \tanh\left(\frac{L(x_2)}{2}\right)\right] = 2\,\mathrm{artanh}\left[\lambda_1 \cdot \lambda_2\right] = L(x_1) \boxplus L(x_2)$$

L-Algebra

- mod-2-sum of 2 statistically independent random variables:

$$L(u_1 \oplus u_2) = 2\,\mathrm{artanh}\left[\tanh\left(\frac{L(x_1)}{2}\right)\cdot\tanh\left(\frac{L(x_2)}{2}\right)\right] = 2\,\mathrm{artanh}\left[\lambda_1 \cdot \lambda_2\right] = L(x_1) \boxplus L(x_2)$$

$$\approx \mathrm{sgn}\left[L(x_1)\right]\cdot\mathrm{sgn}\left[L(x_2)\right]\cdot\min\left\{\left|L(x_1)\right|,\left|L(x_2)\right|\right\}$$



$$L(u_1 \oplus \cdots \oplus u_n) = 2\,\mathrm{artanh}\left[\prod_{i=1}^{n}\tanh\left(L(x_i)/2\right)\right] = \sum_{i=1}^{n}\boxplus\, L(x_i)$$

- mod-2-sum of n variables:

$$\approx \min_i\left\{\left|L(x_i)\right|\right\}\cdot\prod_{i=1}^{n}\mathrm{sgn}\left[L(x_i)\right]$$

General Approach for *Soft-Output* Decoding

- For FEC encoded sequence MAP criterion should be fulfilled
- **Symbol-by-Symbol MAP Criterion**: $L(\hat{u}_i) = \ln \dfrac{p(u_i = 0, \mathbf{y})}{p(u_i = 1, \mathbf{y})}$
  - L-value for estimation of information bit u_i given by receive sequence $\mathbf{y}$
  - Joint probability density function $p(u_i=0/1,\mathbf{y})$ not available → elementary conversions
- Using the completeness, the code space is split into two subsets

$$P(a) = \sum_i P(a, b_i)$$

$\Gamma_i^{(0)}$ = contains all $\mathbf{c}$ with $u_i = 0$

$\Gamma_i^{(1)}$ = contains all $\mathbf{c}$ with $u_i = 1$

$$p(u_i = 0, \mathbf{y}) = \sum_{\mathbf{c} \in \Gamma_i^{(0)}} p(\mathbf{c}, \mathbf{y})$$

$$p(u_i = 1, \mathbf{y}) = \sum_{\mathbf{c} \in \Gamma_i^{(1)}} p(\mathbf{c}, \mathbf{y})$$

$$L(\hat{u}_i) = \ln \frac{\sum_{\mathbf{c} \in \Gamma_i^{(0)}} p(\mathbf{c}, \mathbf{y})}{\sum_{\mathbf{c} \in \Gamma_i^{(1)}} p(\mathbf{c}, \mathbf{y})} = \ln \frac{\sum_{\mathbf{c} \in \Gamma_i^{(0)}} p(\mathbf{y} \mid \mathbf{c}) \cdot \Pr\{\mathbf{c}\}}{\sum_{\mathbf{c} \in \Gamma_i^{(1)}} p(\mathbf{y} \mid \mathbf{c}) \cdot \Pr\{\mathbf{c}\}}$$

sum over $2^k/2=2^{k-1}$ code words in numerator and in denominator

General Approach for *Soft-Output* Decoding

- Assuming statistical independency of the y_j (transmission over AWGNC)
  - Succeeding noise terms n_j are independent, but of course not succeeding code bits c_j (interdependencies introduced by encoder)!
  - $p(\mathbf{y}|\mathbf{c})$ represents probability density conditioned on the hypothesis $\mathbf{c}$
  - y_j are statistically independent random variables

$$p(\mathbf{y}\,|\,\mathbf{c}) = \prod_{j=0}^{n-1} p\left(y_j\,|\,c_j\right)$$

$$L(\hat{u}_i) = \ln \frac{\sum_{\mathbf{c}\in\Gamma_i^{(0)}} \prod_{j=0}^{n-1} p\left(y_j\,|\,c_j\right)\cdot \Pr\{\mathbf{c}\}}{\sum_{\mathbf{c}\in\Gamma_i^{(1)}} \prod_{j=0}^{n-1} p\left(y_j\,|\,c_j\right)\cdot \Pr\{\mathbf{c}\}}$$

- Each codeword $\mathbf{c}$ is uniquely determined by the corresponding info word $\mathbf{u}$ (u_i are statistically independent)

$$\Pr\{\mathbf{c}\} = \Pr\{\mathbf{u}\} = \prod_{j=0}^{k-1} \Pr\{u_j\}$$

$$L(\hat{u}_i) = \ln \frac{\sum_{\mathbf{c}\in\Gamma_i^{(0)}} \prod_{j=0}^{n-1} p\left(y_j\,|\,c_j\right)\cdot \prod_{j=0}^{k-1}\Pr\{u_j\}}{\sum_{\mathbf{c}\in\Gamma_i^{(1)}} \prod_{j=0}^{n-1} p\left(y_j\,|\,c_j\right)\cdot \prod_{j=0}^{k-1}\Pr\{u_j\}}$$

Symbol-by-Symbol MAP

General Approach for *Soft-Output* Decoding

- **Symbol-by-Symbol MAP for systematic encoders**
  - For systematic encoder $u_i = c_i$ holds for $0 \le i \le k\text{-}1$ → i-th term $p(y_i|c_i)$ is constant in numerator and denominator → can be separated together with $P(u_i)$

$$L(\hat{u}_i) = \ln \frac{p(y_i \mid u_i = 0)}{p(y_i \mid u_i = 1)} + \ln \frac{\Pr\{u_i = 0\}}{\Pr\{u_i = 1\}} + \ln \frac{\sum\limits_{\mathbf{c}\in\Gamma_i^{(0)}} \prod\limits_{\substack{j=0\\ j\neq i}}^{n-1} p(y_j \mid c_j) \cdot \prod\limits_{\substack{j=0\\ j\neq i}}^{k-1} \Pr\{u_j\}}{\sum\limits_{\mathbf{c}\in\Gamma_i^{(1)}} \prod\limits_{\substack{j=0\\ j\neq i}}^{n-1} p(y_j \mid c_j) \cdot \prod\limits_{\substack{j=0\\ j\neq i}}^{k-1} \Pr\{u_j\}}$$

$$= L_{ch} \cdot y_i + L_a(u_i) + L_e(u_i)$$

- *Soft-Output* can be split into 3 statistically independent parts:
  - **Systematic part** $L_{ch} \cdot y_i$
  - **A-priori information** $L_a(u_i)$
  - **Extrinsic information** $L_e(u_i)$: information provided by code bits connected with u_i

only for systematic encoders

General Approach for *Soft-Output* Decoding

- Compact description of extrinsic information

$$\prod_{\substack{j=0\\j\neq i}}^{n-1} p\left(y_j \mid c_j\right)\cdot\prod_{\substack{j=0\\j\neq i}}^{k-1}\Pr\left\{u_j\right\}=\prod_{\substack{j=0\\j\neq i}}^{n-1} p\left(y_j;c_j\right) \quad \text{with} \quad p(y_\ell;c_\ell)=\begin{cases} p\left(y_\ell \mid c_\ell\right)\cdot\Pr\left\{u_\ell\right\} & 0\leq \ell<k \\ p\left(y_\ell \mid c_\ell\right) & k\leq \ell<n \end{cases}$$

$$\Rightarrow \quad L_e\left(\hat{u}_i\right)=\ln\frac{\sum\limits_{\mathbf{c}\in\Gamma_i^{(0)}}\prod\limits_{\substack{j=0\\j\neq i}}^{n-1} p\left(y_j \mid c_j\right)\cdot\prod\limits_{\substack{j=0\\j\neq i}}^{k-1}\Pr\left\{c_j\right\}}{\sum\limits_{\mathbf{c}\in\Gamma_i^{(1)}}\prod\limits_{\substack{j=0\\j\neq i}}^{n-1} p\left(y_j \mid c_j\right)\cdot\prod\limits_{\substack{j=0\\j\neq i}}^{k-1}\Pr\left\{c_j\right\}}=\ln\frac{\sum\limits_{\mathbf{c}\in\Gamma_i^{(0)}}\prod\limits_{\substack{j=0\\j\neq i}}^{n-1} p\left(y_j;c_j\right)}{\sum\limits_{\mathbf{c}\in\Gamma_i^{(1)}}\prod\limits_{\substack{j=0\\j\neq i}}^{n-1} p\left(y_j;c_j\right)}$$

- Calculation of extrinsic information with LLR‘s:

$$L_e\left(\hat{u}_i\right)=\ln\frac{\sum\limits_{\mathbf{c}\in\Gamma_i^{(0)}}\prod\limits_{\substack{j=0\\j\neq i}}^{n-1}\exp\left[-L(c_j;y_j)\cdot c_j\right]}{\sum\limits_{\mathbf{c}\in\Gamma_i^{(1)}}\prod\limits_{\substack{j=0\\j\neq i}}^{n-1}\exp\left[-L(c_j;y_j)\cdot c_j\right]} \quad \text{with} \quad L(c_\ell;y_\ell)=\begin{cases} L_{ch}\cdot y_\ell+L_a(u_\ell) & 0\leq \ell<k \\ L_{ch}\cdot y_\ell & k\leq \ell<n \end{cases}$$

Soft-Output Decoding of Repetition Codes

- Code word $\mathbf{c} = [c_0\ c_1 \cdots c_{n-1}]$ contains n repetitions of information word $\mathbf{u} = [u_0]$
  - Set of all code words for $n = 3$ is given by $\Gamma = \{000, 111\}$

$$L(\hat{u}_0) = \ln \frac{\sum\limits_{\mathbf{c}\in\Gamma_0^{(0)}} \prod\limits_{j=0}^{n-1} p(y_j \mid c_j)\cdot \Pr\{\mathbf{c}\}}{\sum\limits_{\mathbf{c}\in\Gamma_0^{(1)}} \prod\limits_{j=0}^{n-1} p(y_j \mid c_j)\cdot \Pr\{\mathbf{c}\}} = \ln \frac{\prod\limits_{j=0}^{n-1} p(y_j \mid 0)\cdot \Pr\{\mathbf{c}=[000]\}}{\prod\limits_{j=0}^{n-1} p(y_j \mid 1)\cdot \Pr\{\mathbf{c}=[111]\}}$$

$$= \ln \frac{p(y_0 \mid 0)\cdot p(y_1 \mid 0)\cdot p(y_2 \mid 0)\cdot \Pr\{u_i = 0\}}{p(y_0 \mid 1)\cdot p(y_1 \mid 1)\cdot p(y_2 \mid 1)\cdot \Pr\{u_i = 1\}}$$

$$= \ln \frac{p(y_0 \mid 0)}{p(y_0 \mid 1)} + \ln \frac{p(y_1 \mid 0)}{p(y_1 \mid 1)} + \ln \frac{p(y_2 \mid 0)}{p(y_2 \mid 1)} + \ln \frac{\Pr\{u_i = 0\}}{\Pr\{u_i = 0\}}$$

$$= L(y_0 \mid c_0) + L(y_1 \mid c_1) + L(y_2 \mid c_2) + L_a(u_0)$$

  - Corresponds to averaging of LLRs

Soft-Output Decoding using the Dual Code

- Calculation of extrinsic information requires summation over all code words $\mathbf{c}$ of the code space Γ
  - The (255,247,3) Hamming code contains $2^{247} = 2.3 \cdot 10^{74}$ code words
- Instead of calculating the LLR over all code words $\mathbf{c}$ of the code $\mathcal{C}$, it is also possible to perform this calculation with respect to the dual code $\mathcal{C}^{\perp}$
  - Beneficial, if the number of parity bits is relatively small
    → dual code for (255,247,3) Hamming code contains only $2^8 = 256$ code words
- Calculation of extrinsic information with dual code:

$$L_e(\hat{u}_i) = \ln \frac{\sum_{\mathbf{c}' \in \Gamma^{\perp}} \prod_{\substack{\ell=0 \\ \ell \neq i}}^{n-1} \left[\tanh\left(\frac{L(c_\ell; y_\ell)}{2} \right) \right]^{c'_\ell}}{\sum_{\mathbf{c}' \in \Gamma^{\perp}} (-1)^{c'_i} \prod_{\substack{\ell=0 \\ \ell \neq i}}^{n-1} \left[\tanh\left(\frac{L(c_\ell; y_\ell)}{2} \right) \right]^{c'_\ell}}$$

Summation over all 2^{n-k} code words $\mathbf{c}$’ of the dual code

Soft-Output Decoding of (4,3,2)-SPC using the Dual Code

- Calculation of extrinsic information requires summation over $2^3 = 8$ code words. Instead, the dual code contains only $2^{n-k} = 2$ words $\Gamma^{\perp} = \{0000, 1111\}$.
- Calculation of LLR

$$L(\hat{u}_i) = L_{ch} \cdot y_i + \ln \frac{1 + \prod_{\substack{\ell=0 \\ \ell \neq i}}^{n-1} \left[\tanh\left(\frac{L(c_\ell; y_\ell)}{2} \right) \right]}{1 - \prod_{\substack{\ell=0 \\ \ell \neq i}}^{n-1} \left[\tanh\left(\frac{L(c_\ell; y_\ell)}{2} \right) \right]}$$

$$= L_{ch} \cdot y_i + 2\,\mathrm{artanh}\left(\prod_{\substack{\ell=0 \\ \ell \neq i}}^{n-1} \left[\tanh\left(\frac{L(c_\ell; y_\ell)}{2} \right) \right] \right)$$

$$\approx L_{ch} \cdot y_i + \min_{\ell \neq i} \left\{ \left| L(c_\ell; y_\ell) \right| \right\} \cdot \prod_{\substack{\ell=0 \\ \ell \neq i}}^{n-1} \mathrm{sgn}\left[L(c_\ell; y_\ell) \right]$$

First term in numerator and denominator ($\mathbf{c}$=0000) is one.

with $$\ln \frac{1+x}{1-x} = 2\,\mathrm{artanh}(x)$$

Each $\mathbf{c} \in \Gamma$ fulfills $\mathbf{c}\mathbf{c}^T = 0$, i.e. c_i is given by modulo-2-sum of all other code bits c_j:

$$c_i = \sum_{j \neq i} c_j \quad \Rightarrow \quad L_e(c_i) = \sum_{\substack{j=1 \\ j \neq i}}^{n} \boxed{+}\, L(x_j)$$

Soft-Output Decoding for (4,3,2)-SPC-Code

$$L_{ch} = 4 \cdot EsN0_{\text{lin}} = 4 \cdot 10^{EsN0_{\text{dB}}/10} = 4 \cdot 10^{2/10} = 6{,}34$$

$E_s/N_0 = 2\text{dB}$

u

1	0	1

encoding

c

1	0	1	0

BPSK

x

-1	+1	-1	+1

AWGN

y

-0.8	+1.1	+0.3	+0.4

HD →

-1	+1	+1	+1

error detected.
but not corrected

$L_{ch} \cdot \mathbf{y}$

-5.1	+7.0	+1.9	+2.5

+ Approximation for $L_e(c_i) = \sum_{\substack{j=1 \\ j \neq i}}^{n} \boxed{+} L(x_j)$

$L_e(\hat{\mathbf{u}})$

+1.9	-1.9	-2.5	-1.9

= $L_{ch} \cdot \mathbf{y} + L_e(\hat{\mathbf{u}})$

$L(\hat{\mathbf{u}})$

-3.2	+5.1	-0.6	+0.6

$\Pr\{\hat{\mathbf{u}} \text{ correct}\}$

0.96	0.99	0.65	0.65

HD →

-1	+1	-1	+1

error corrected

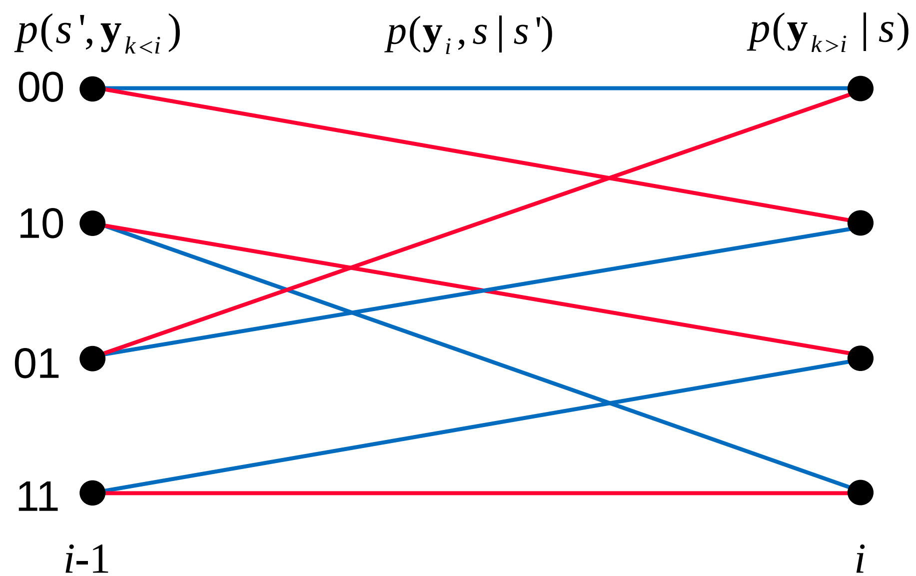
BCJR Algorithm for Convolutional Codes

- ***Symbol-by-Symbol* MAP Decoding**: Bahl, Cocke, Jelinek, Raviv (1972)

$$L(\hat{u}_i) = \ln \frac{p(u_i = 0, \mathbf{y})}{p(u_i = 1, \mathbf{y})} = \ln \frac{\sum_{(s',s),\, u_i=0} p(s',s,\mathbf{y})}{\sum_{(s',s),\, u_i=1} p(s',s,\mathbf{y})} = \ln \frac{\sum_{(s',s),\, u_i=0} p(s',s,\mathbf{y}_{k<i},\mathbf{y}_i,\mathbf{y}_{k>i})}{\sum_{(s',s),\, u_i=1} p(s',s,\mathbf{y}_{k<i},\mathbf{y}_i,\mathbf{y}_{k>i})}$$

  - Efficient calculation of LLR based on the Trellis diagram (exploiting Markov prop.)

state s' state s



Trellis of a RSC with L_c=3

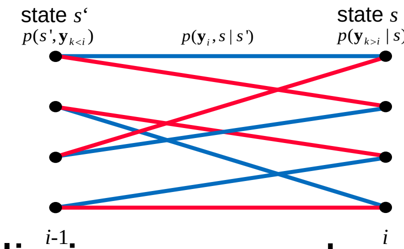
$u_i = 1$

$u_i = 0$

$$\mathbf{y} = [\mathbf{y}_1\ \mathbf{y}_2 \ldots \mathbf{y}_N]$$

$$\mathbf{y}_i = [y_{i,0}\ y_{i,1} \ldots y_{i,n-1}]$$

BCJR Algorithm for Convolutional Codes



- Splitting up the observations $\mathbf{y}_{k>i}$

$$p(s',s,\mathbf{y}_{k<i},\mathbf{y}_i,\mathbf{y}_{k>i}) = p(\mathbf{y}_{k>i}\,|\,s',s,\mathbf{y}_{k<i},\mathbf{y}_i)\cdot p(s',s,\mathbf{y}_{k<i},\mathbf{y}_i)$$

- **Backward probability**: Probability of the sequence $\mathbf{y}_{k>i}$, if the trellis is assumed in state s at time instant i

$$\beta_i(s) = p(\mathbf{y}_{k>i}\,|\,s',s,\mathbf{y}_{k<i},\mathbf{y}_i) = p(\mathbf{y}_{k>i}\,|\,s)$$

If state s at time instant i is known, the parameter s', $\mathbf{y}_i$, $\mathbf{y}_{k<i}$ are not relevant

- Splitting up the observations $\mathbf{y}_i$

$$p(s',s,\mathbf{y}_{k<i},\mathbf{y}_i) = p(s,\mathbf{y}_i\,|\,s',\mathbf{y}_{k<i})\cdot p(s',\mathbf{y}_{k<i})$$

- **Transition probability**: Probability of observing $\mathbf{y}_i$ under the condition that the transition from s' to s takes place at time instant i → $\mathbf{y}_{k<i}$ not relevant

$$\gamma_i(s',s) = p(s,\mathbf{y}_i\,|\,s',\mathbf{y}_{k<i}) = p(s,\mathbf{y}_i\,|\,s')$$

$$= \frac{p(s',s,\mathbf{y}_i)}{\Pr\{s'\}} = p(\mathbf{y}_i\,|\,s',s)\frac{\Pr\{s',s\}}{\Pr\{s'\}} = p(\mathbf{y}_i\,|\,s',s)\cdot\Pr\{s\,|\,s'\}$$

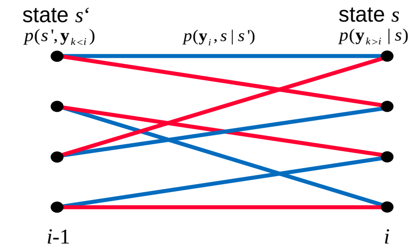
$p\{\mathbf{y}_i|s',s\}$: transition probability of channel
$\Pr\{s|s'\}$: a-priori-information

  - Possibility to use **a-priori knowledge** within the decoding process → $\Pr\{s|s'\} \sim u_i$

BCJR Algorithm for Convolutional Codes

- **Forward probability**: $\alpha_{i-1}(s') = p(s', \mathbf{y}_{k<i})$

Probability of sequence $\mathbf{y}_{k<i}$, if the trellis is assumed in state s' at time instant i-1



- Probability density splits into three terms

$$p(s', s, \mathbf{y}_{k<i}, \mathbf{y}_i, \mathbf{y}_{k>i}) = \alpha_{i-1}(s') \cdot \gamma_i(s', s) \cdot \beta_i(s)$$

- Compact description of Symbol-by-Symbol MAP

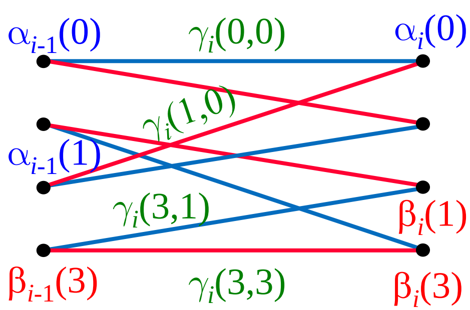
$$L(\hat{u}_i) = \ln \frac{\sum_{(s',s),\, u_i=0} p(s', s, \mathbf{y}_{k<i}, \mathbf{y}_i, \mathbf{y}_{k>i})}{\sum_{(s',s),\, u_i=1} p(s', s, \mathbf{y}_{k<i}, \mathbf{y}_i, \mathbf{y}_{k>i})} = \ln \frac{\sum_{(s',s),\, u_i=0} \alpha_{i-1}(s') \cdot \gamma_i(s', s) \cdot \beta_i(s)}{\sum_{(s',s),\, u_i=1} \alpha_{i-1}(s') \cdot \gamma_i(s', s) \cdot \beta_i(s)}$$

- Recursive Calculation
  - Forward probability

$$\alpha_i(s) = p(s, \mathbf{y}_{k<i+1}) = \sum_{s'} \gamma_i(s', s) \cdot \alpha_{i-1}(s')$$

  - Backward probability

$$\beta_{i-1}(s') = p(\mathbf{y}_{k>i-1} \mid s') = \sum_{s} \gamma_i(s', s) \cdot \beta_i(s)$$



  - Initialization

Terminated code

$$\alpha_0(s') = \begin{cases} 1 & s' = 0 \\ 0 & s' \neq 0 \end{cases} \qquad \beta_N(s) = \begin{cases} 1 & s' = 0 \\ 0 & s' \neq 0 \end{cases}$$

otherwise

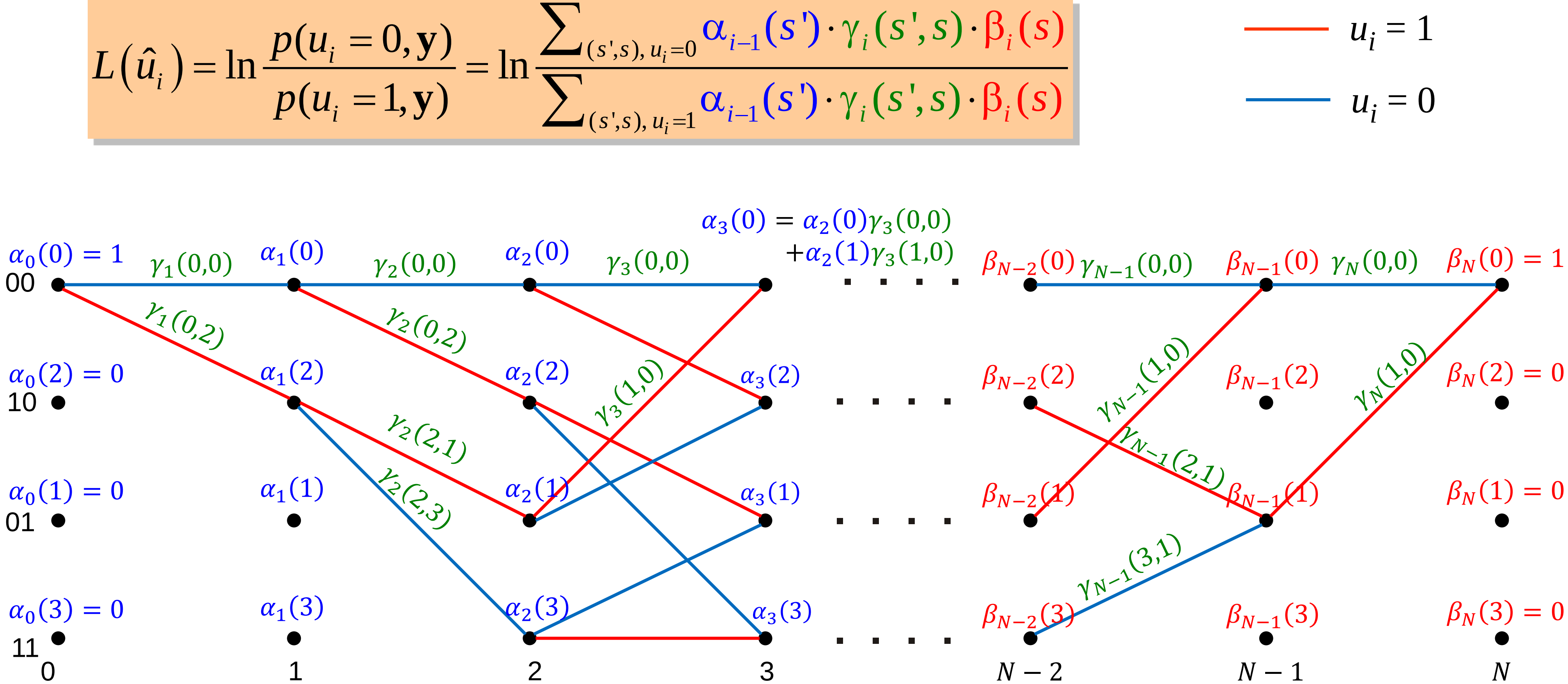
$$\beta_N(s) = 2^{-m}$$

(m memory elements)

BCJR Algorithm for Convolutional Codes

- *Symbol-by-Symbol* MAP Decoding:

$$L(\hat{u}_i) = \ln \frac{p(u_i = 0, \mathbf{y})}{p(u_i = 1, \mathbf{y})} = \ln \frac{\sum_{(s',s),\, u_i=0} \alpha_{i-1}(s') \cdot \gamma_i(s',s) \cdot \beta_i(s)}{\sum_{(s',s),\, u_i=1} \alpha_{i-1}(s') \cdot \gamma_i(s',s) \cdot \beta_i(s)}$$



Calculation in Logarithmic Domain

- Implementation with respect to probabilities is complicated
  → numerical problems → implementation in the logarithmic domain favorable
  - Transition variable $$\bar{\gamma}_i(s',s) = \ln \gamma_i(s',s) = \ln p(\mathbf{y}_i | s',s) + \ln \Pr\{s | s'\}$$ $$= C - \frac{1}{2\sigma_N^2}\|\mathbf{y}_i - \mathbf{x}(s',s)\|^2 + \ln \Pr\{u_i = u(s',s)\}$$
  - Forward variable $$\bar{\alpha}_i(s) = \ln \alpha_i(s) = \ln\left(\sum_{s'} \gamma_i(s',s)\cdot\alpha_{i-1}(s')\right) = \ln\left(\sum_{s'} \exp\left(\bar{\gamma}_i(s',s) + \bar{\alpha}_{i-1}(s')\right)\right)$$
  - Backward variable $$\bar{\beta}_{i-1}(s') = \ln \beta_{i-1}(s') = \ln\left(\sum_{s} \gamma_i(s',s)\cdot\beta_i(s)\right) = \ln\left(\sum_{s} \exp\left(\bar{\gamma}_i(s',s) + \bar{\beta}_i(s)\right)\right)$$
  - Initialization $$\bar{\alpha}_0(s') = \begin{cases} 0 & s' = 0 \\ -\infty & s' \neq 0 \end{cases}$$
    Terminated code $$\bar{\beta}_N(s) = \begin{cases} 0 & s' = 0 \\ -\infty & s' \neq 0 \end{cases}$$
    otherwise $$\bar{\beta}_N(s) = const.$$

Calculation in Logarithmic Domain: Jacobi Logarithm

- In recursion, ln of sum of exponents occur

$$\ln\left(e^{x_1}+e^{x_2}\right)=\max\left[x_1,x_2\right]+\ln\left(1+e^{-|x_1-x_2|}\right)=\max{}^*\left[x_1,x_2\right]$$

- Proof
  - For $x_1 > x_2$

$$\max{}^*\left[x_1,x_2\right]=\ln\left(e^{x_1}\left(1+e^{-(x_1-x_2)}\right)\right)=\ln\left(e^{x_1}\right)+\ln\left(1+e^{-(x_1-x_2)}\right)=x_1+\ln\left(1+e^{-|x_1-x_2|}\right)$$

  - For $x_1 \le x_2$

$$\max{}^*\left[x_1,x_2\right]=\ln\left(e^{x_2}\left(1+e^{-(x_2-x_1)}\right)\right)=\ln\left(e^{x_2}\right)+\ln\left(1+e^{-(x_2-x_1)}\right)=x_2+\ln\left(1+e^{-|x_1-x_2|}\right)$$

  - Second term has small range between 0 and ln 2
    → efficiently be implemented by a lookup table w.r.t $|x_1-x_2|$

Calculation in Logarithmic Domain: Jacobi Logarithm

- Simplify logarithm of sums $\ln\left(e^{x_1}+e^{x_2}\right)=\max{}^*\left[x_1,x_2\right]=\max\left[x_1,x_2\right]+\ln\left(1+e^{-|x_1-x_2|}\right)$
  - Forward variable

$$\begin{aligned}\bar{\alpha}_i(s)=\ln\alpha_i(s)&=\ln\left(\sum_{s'}\exp\left(\bar{\gamma}_i(s',s)+\bar{\alpha}_{i-1}(s')\right)\right)\\&=\max{}^*\left[\bar{\gamma}_i(s_1',s)+\bar{\alpha}_{i-1}(s_1'),\bar{\gamma}_i(s_2',s)+\bar{\alpha}_{i-1}(s_2')\right]\\&=\max_{s'}\left[\bar{\gamma}_i(s',s)+\bar{\alpha}_{i-1}(s')\right]+\underbrace{\ln\left(1+e^{-|\Delta_i|}\right)}\end{aligned}$$

correction term

$$\begin{aligned}\Delta_i&=\left(\bar{\gamma}_i(s_1',s)+\bar{\alpha}_{i-1}(s_1')\right)\\&\quad-\left(\bar{\gamma}_i(s_2',s)+\bar{\alpha}_{i-1}(s_2')\right)\end{aligned}$$

  - Backward variable

$$\begin{aligned}\bar{\beta}_{i-1}(s')=\ln\beta_{i-1}(s')&=\max{}^*\left[\bar{\gamma}_i(s',s_1)+\bar{\beta}_i(s_1),\bar{\gamma}_i(s',s_2)+\bar{\beta}_i(s_2)\right]\\&=\max_{s}\left[\bar{\gamma}_i(s',s)+\bar{\beta}_i(s)\right]+\underbrace{\ln\left(1+e^{-|\Delta_i|}\right)}\end{aligned}$$

correction term

$$\begin{aligned}\Delta_i&=\left(\bar{\gamma}_i(s',s_1)+\bar{\beta}_i(s_1)\right)\\&\quad-\left(\bar{\gamma}_i(s',s_2)+\bar{\beta}_i(s_2)\right)\end{aligned}$$

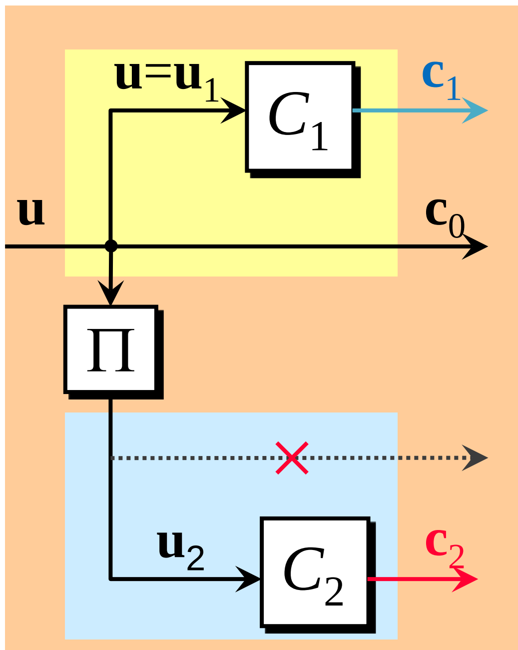
- Declaration:
  - **Log-MAP**: implementation of BCJR in log-domain **with** correction term
  - **Max-Log-MAP**: implementation in log-domain **without** correction term

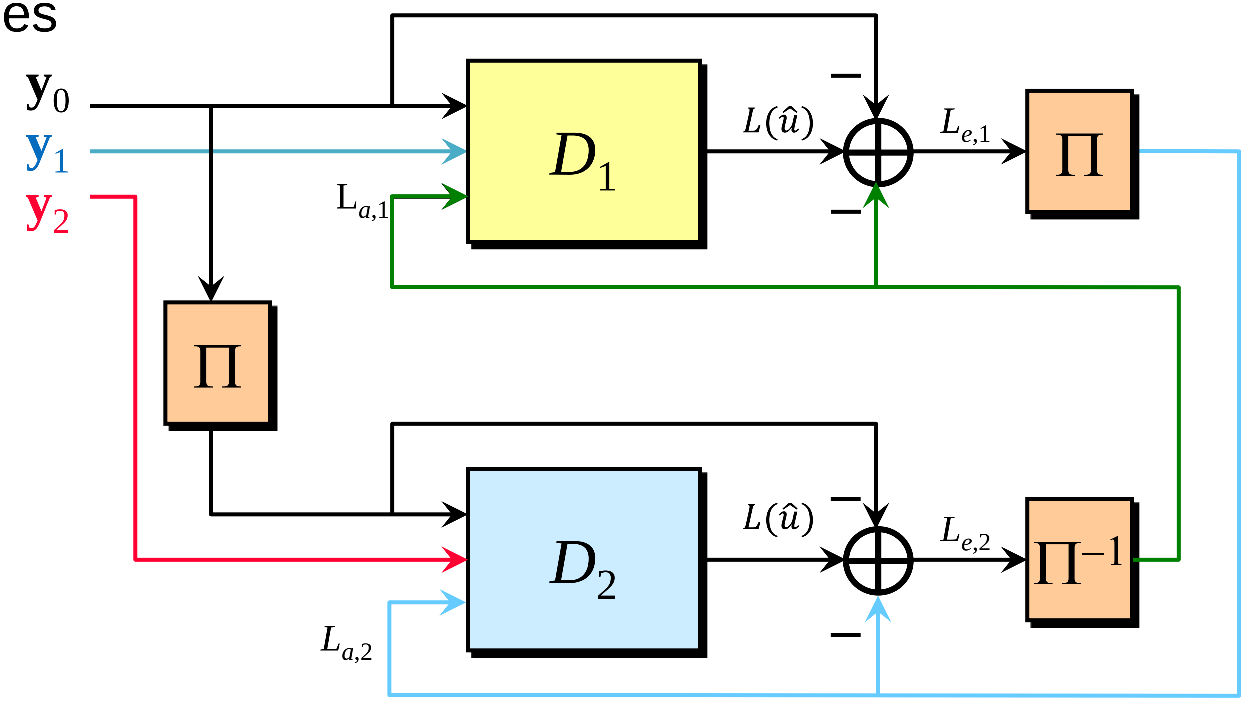
Iterative Decoding

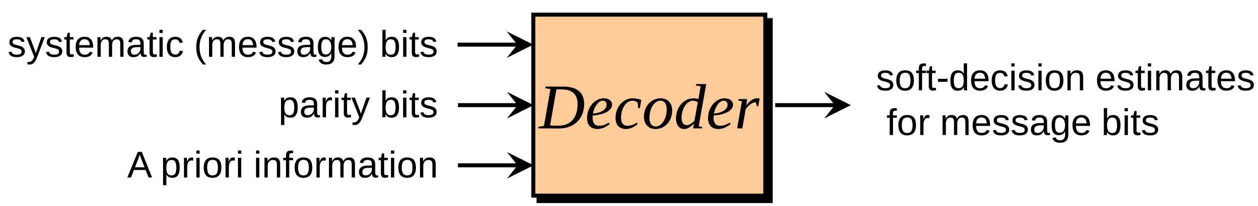
- General Structure for Parallel Concatenated Codes
- Turbo Decoding for (24,16,3)-Product Code
- Simulation Results
- Turbo Decoding for Serially Concatenated Codes

General Concept for Iterative Decoding

- Parallel Concatenated Codes







Turbo Decoding for (24,16,3) Modified Product Code (1)

$\mathbf{u}$

1	0	0	1
0	1	1	1
1	0	1	0
0	0	0	1

encoding → BPSK

$\mathbf{x}$

-1	+1	+1	-1	+1
+1	-1	-1	-1	-1
-1	+1	-1	+1	+1
+1	+1	+1	-1	-1
+1	-1	+1	-1	

AWGN → SNR=2 dB

LLR $L_{ch} \cdot \mathbf{y}$

0.6	7.6	1.3	-3.2	6.3
5.1	-4.4	3.8	-0.6	-9.5
-7.6	3.2	-5.7	7.6	1.3
1.3	-1.3	8.2	-9.5	-12.7
1.9	-5.7	7.6	-7.0	

1. vertical decoding → extrinsic information

$L^{|}_{e,1}(\hat{\mathbf{u}})$

-1.3	-1.3	-3.8	-0.6
-0.6	1.3	-1.3	-3.2
0.6	-1.3	1.3	0.6
-0.6	3.2	-1.3	-0.6

$L_{ch} \cdot \mathbf{y} + L^{|}_{e,1}(\hat{\mathbf{u}})$

$$L^{|}_{1}(\hat{\mathbf{u}}) = L_{ch} \cdot \mathbf{y} + L^{|}_{e,1}(\hat{\mathbf{u}})$$

-0.7	6.3	-2.5	-3.8
4.5	-3.1	2.5	-3.8
-7.0	1.9	-4.4	8.2
0.7	1.9	6.9	-10.1

Vertical extrinsic info serves as horizontal a-priori info

$$L^{-}_{a,1}(\hat{\mathbf{u}}) = L^{|}_{e,1}(\hat{\mathbf{u}})$$

-1.3	-1.3	-3.8	-0.6
-0.6	1.3	-1.3	-3.2
0.6	-1.3	1.3	0.6
-0.6	3.2	-1.3	-0.6

Turbo Decoding for (24,16,3) Modified Product Code (2)

$L_{ch} \cdot \mathbf{y} + L^{-}_{a,1}(\hat{\mathbf{u}})$

-0.7	6.3	-2.5	-3.8	6.3
4.5	-3.1	2.5	-3.8	-9.5
-7.0	1.9	-4.4	8.2	1.3
0.7	1.9	6.9	-10.1	-12.7
1.9	-5.7	7.6	-7.0	

1. horizontal decoding

$L^{-}_{e,1}(\hat{\mathbf{u}})$

2.5	-0.7	0.7	0.7
-2.5	2.5	-3.1	2.5
-1.3	1.3	-1.3	1.3
1.9	0.7	0.7	-0.7

$L_{ch} \cdot \mathbf{y} + L^{-}_{e,1}(\hat{\mathbf{u}}) + L^{-}_{a,1}(\hat{\mathbf{u}})$

$L^{-}_{1}(\hat{\mathbf{u}})$

1.8	5.6	-1.8	-3.1
2.0	-0.6	-0.6	-1.3
-8.3	3.2	-5.7	9.5
2.6	2.6	7.6	-10.8

$L^{-}_{e,1}(\hat{\mathbf{u}}) = L^{|}_{a,2}(\hat{\mathbf{u}})$

$L_{ch} \cdot \mathbf{y} + L^{|}_{a,2}(\hat{\mathbf{u}})$

3.1	6.9	2.1	-2.5	6.3
2.6	-1.9	0.7	1.9	-9.5
-8.9	4.5	-7.0	8.9	1.3
3.2	-0.6	8.9	-10.2	-12.7
1.9	-5.7	7.6	-7.0	

$\hat{\mathbf{u}}_1$

0	0	1	1
0	1	1	1
1	0	1	0
0	0	0	1

Turbo Decoding for (24,16,3) Modified Product Code (3)

$L_{ch}\,\mathbf{y} + L^{|}_{a,2}(\mathbf{u})$

3.1	6.9	2.1	-2.5	6.3
2.6	-1.9	0.7	1.9	-9.5
-8.9	4.5	-7.0	8.9	1.3
3.2	-0.6	8.9	-10.2	-12.7
1.9	-5.7	7.6	-7.0	

$\hat{\mathbf{u}}_2$

1	0	0	1
0	1	1	1
1	0	1	0
x	x	0	1

$L^{-}_{2}(\hat{\mathbf{u}})$

-1.9	7.6	2.1	-1.9
1.5	-2.1	-1.4	1.4
-7.0	3.9	-6.3	1.3
0	0	6.9	0.6

2. vertical decoding

$L^{|}_{e,2}(\hat{\mathbf{u}})$

-1.9	-0.6	-0.7	1.9
-1.9	0.6	-2.1	-2.5
1.9	-0.6	0.7	-1.9
-1.9	1.9	-0.7	1.9

$L_{ch} \cdot \mathbf{y} + L^{|}_{e,2}(\hat{\mathbf{u}}) + L^{|}_{a,2}(\hat{\mathbf{u}})$

$L_{ch}\,\mathbf{y} + L^{-}_{a,2}(\mathbf{u})$

$L_2^{|}(\hat{\mathbf{u}})$

-1.3	7.0	0.6	-1.3	6.3
3.2 1.2	6.3	1.4	-3.6 0.6	-9.5
0.7	-1.3	-1.4	-0.6	1.3
-7.0	3.9	-6.3	7.0	-12.7
1.3	1.3	8.2	-8.3	

2. horizontal decoding

$L^{-}_{e,2}(\hat{\mathbf{u}})$

-0.6	0.6	1.3	-0.6
-1.7	1.7	-3.1	1.7
-1.3	1.3	-1.3	1.3
0.6	-0.6	-0.6	0.6

$L_{ch} \cdot \mathbf{y} + L^{-}_{e,2}(\hat{\mathbf{u}}) + L^{-}_{a,2}(\hat{\mathbf{u}})$

Turbo Decoding for (24,16,3) Modified Product Code (4)

$L_{ch}\,\mathbf{y} + L^{|}_{a,3}\,(\mathbf{u})$

0.0	8.2	2.6	-3.8	6.3
3.4	-2.7	0.7	1.1	-9.5
-8.9	4.5	-7.0	8.9	1.3
1.9	-1.9	8.2	-8.9	-12.7
1.9	-5.7	7.6	-7.0	

$\hat{\mathbf{u}}_3$

1	0	0	1
0	1	1	1
1	0	1	0
0	0	0	1

$L^{-}_{2}\,(\hat{\mathbf{u}})$

-1.9	6.3	1.9	-2.7
3.9	-1.3	-1.3	-3.2
-8.9	2.6	-6.3	8.8
2.7	2.7	8.8	-9.7

3. vertical decoding

$L^{|}_{e,3}\,(\hat{\mathbf{u}})$

-1.9	-1.9	-0.7	1.1
0	1.9	-2.6	-3.8
0	-1.9	0.7	-1.1
0	2.7	-0.7	1.1

$L_{ch}\cdot\mathbf{y} + L^{|}_{e,3}\,(\hat{\mathbf{u}}) + L^{|}_{a,3}\,(\hat{\mathbf{u}})$

$L_{ch}\,\mathbf{y} + L^{-}_{a,3}\,(\mathbf{u})$

$L_3^{|}(\hat{\mathbf{u}})$

-1.3	5.7	0.6	-2.1	6.3
5.1	-2.5	1.2	-4.4	-9.5
-7.6	1.3	-5.0	7.5	1.3
1.3	1.4	7.5	-8.4	-12.7
1.9	-5.7	7.6	-7.0	

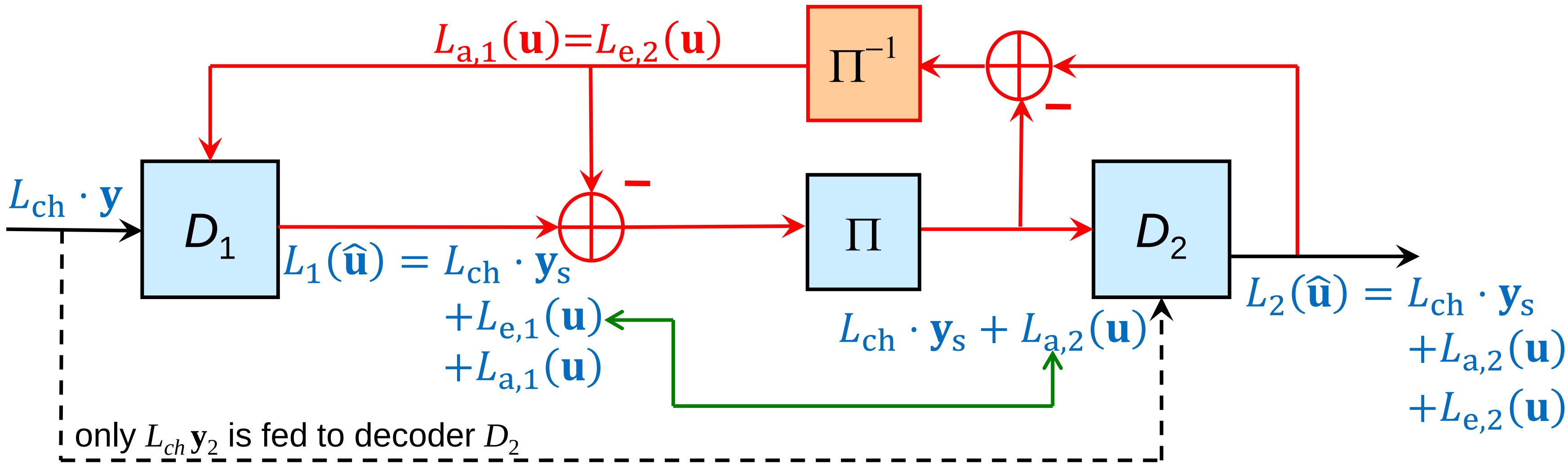
3. horizontal decoding

$L^{-}_{e,3}\,(\hat{\mathbf{u}})$

-0.6	0.6	1.3	-0.6
-1.2	1.2	-2.5	1.2
-1.3	1.3	-1.3	1.3
1.4	1.3	1.3	-1.3

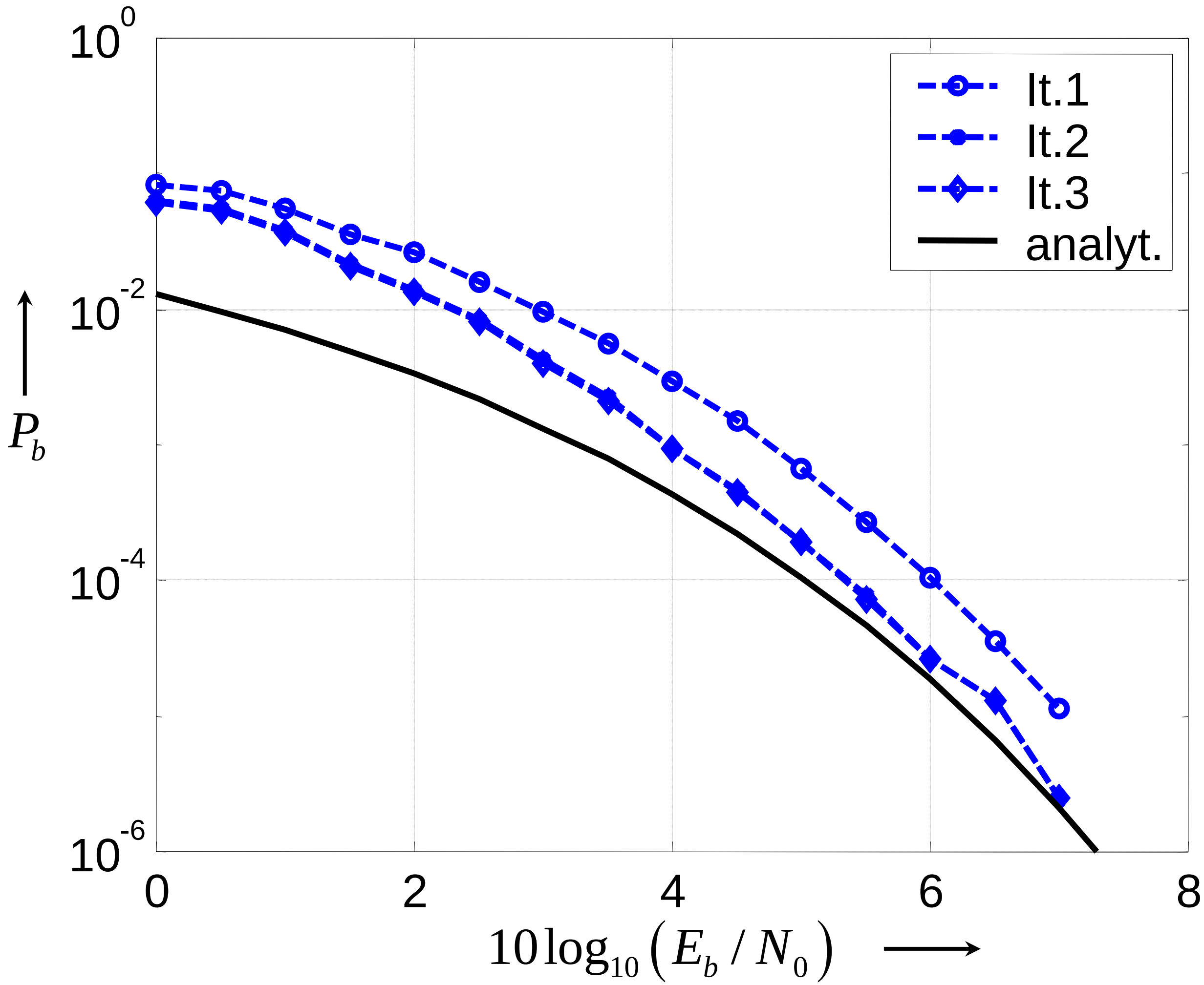
$L_{ch}\cdot\mathbf{y} + L^{-}_{e,3}\,(\hat{\mathbf{u}}) + L^{-}_{a,3}\,(\hat{\mathbf{u}})$

Turbo Decoding for Parallel Concatenated Codes



- Both decoders estimate same information word $\mathbf{u}$ and each decoder receives corresponding channel outputs
- Systematic information bits $\mathbf{y}_s$ are fed to D_2 via D_1 and Π
- Each decoder generates **extrinsic information** for bit $\mathbf{u}$ serving as a priori LLRs for other decoder
- A priori LLRs improve decoders' performance in each iteration as long as they are statistically independent of regular inputs

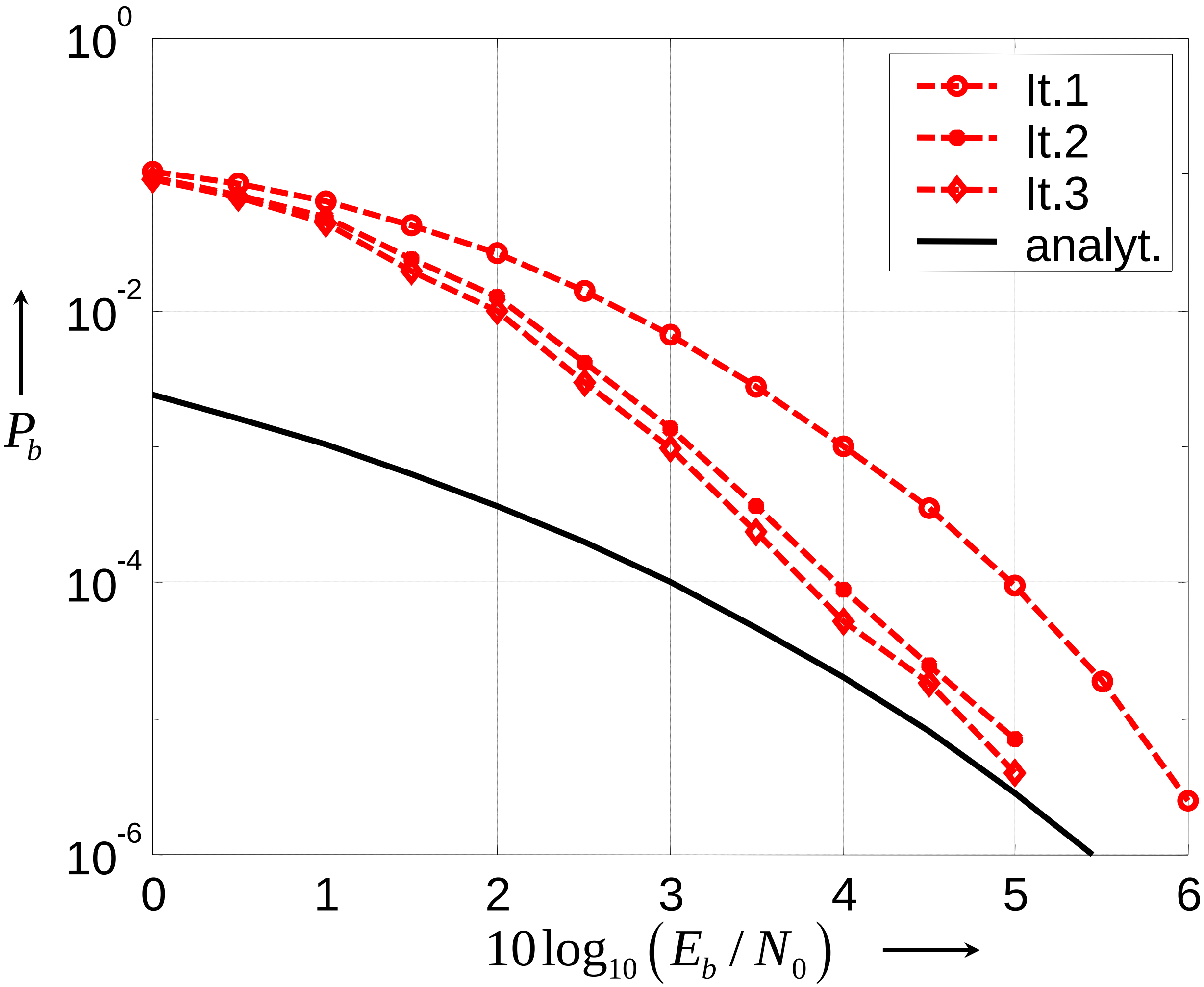
Simulation Results for Modified Product Codes (7,4,3)-Hamming Codes



- Observations
  - Gains decrease with number of iterations
  - Same info bits are estimated and correlation of a-priori information increases
  - With the larger interleaver length the gains of subsequent iterations are generally larger → statistical independence of bits is required

Simulation Results for Modified Product Codes

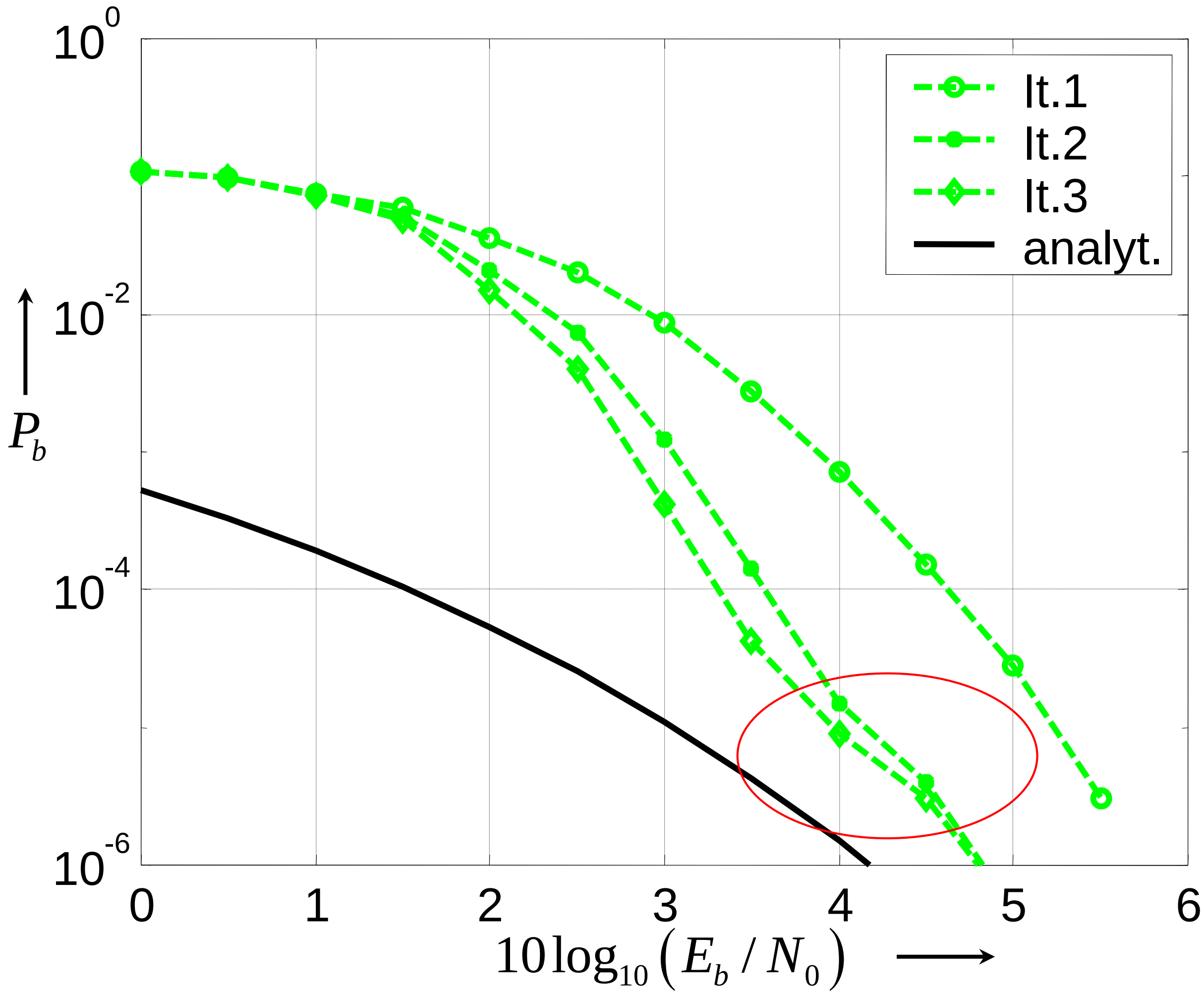
(15,11,3)-Hamming-Codes



- Observations
  - Larger interleaver leads to improved statistic
    → gains for iteration 3

Simulation Results for Modified Product Codes

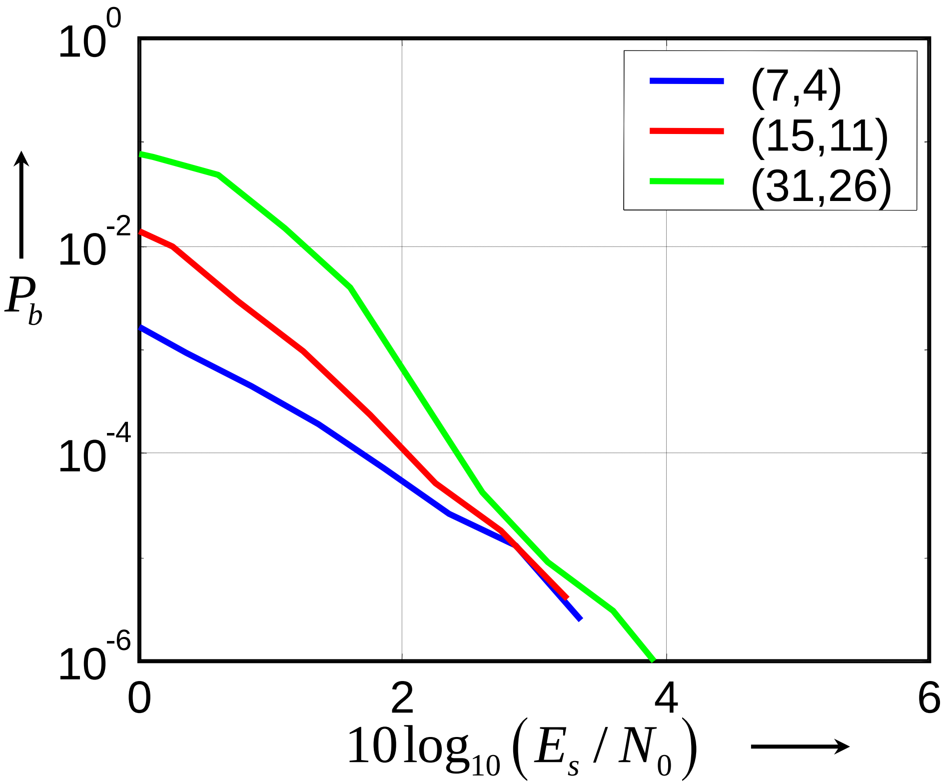
(31,26,3)-Hamming-Codes

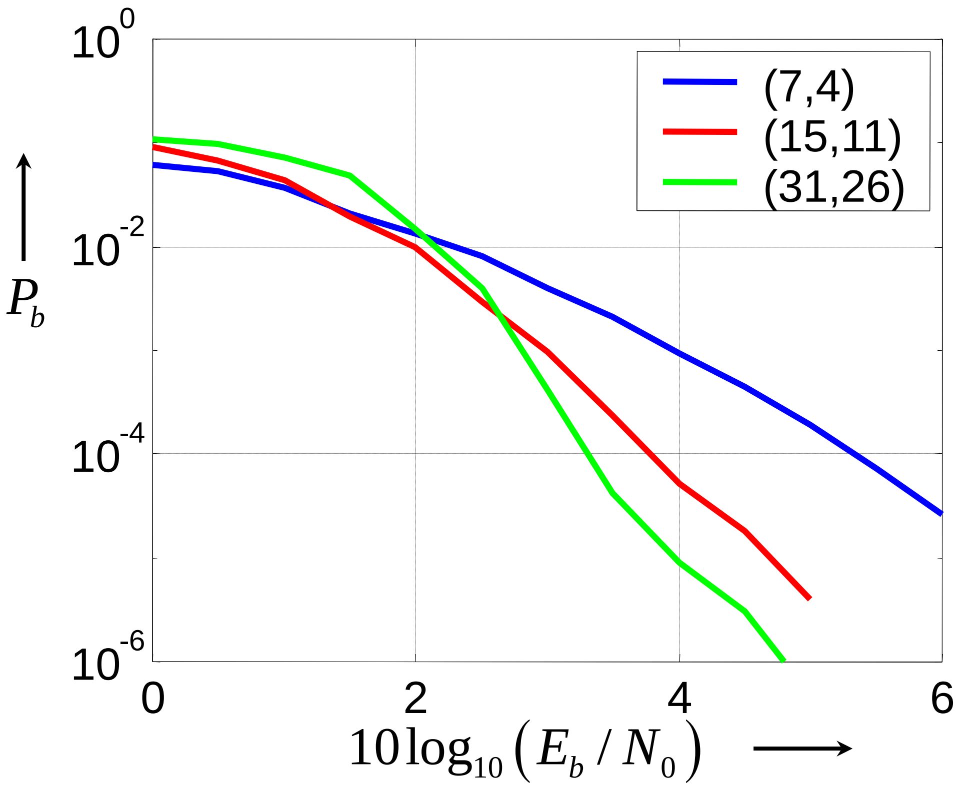


- Observations
  - Larger interleaver leads to improved statistic
    → gains for iteration 3
  - For larger SNR the BER flattens
    → minimum distance dominates error rate for large SNR

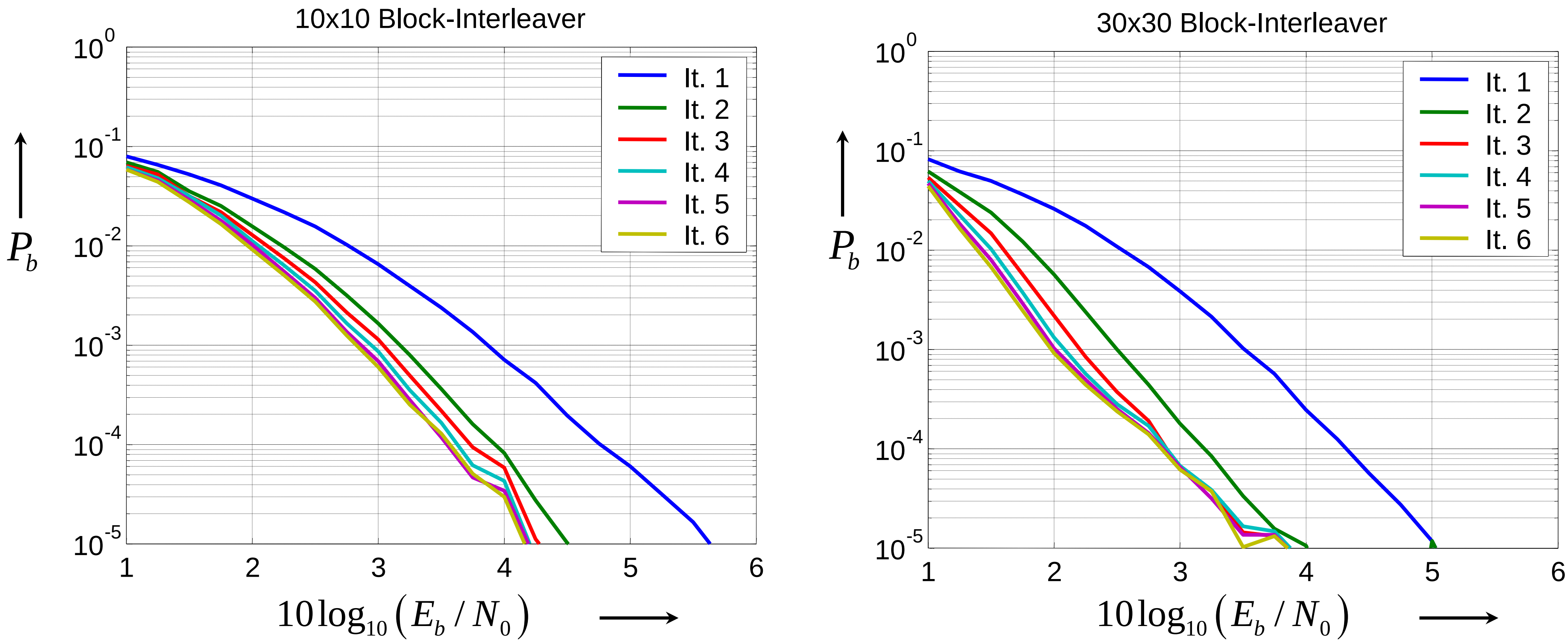
Simulation Results for Modified Product Codes

- Hamming codes have $d_{\min} = 3$ for all lengths n
  - Analyzed product codes have same $d_{\min}$ → similar error rates versus E_s/N_0
  - Code rates are different → longer product codes are better versus E_b/N_0



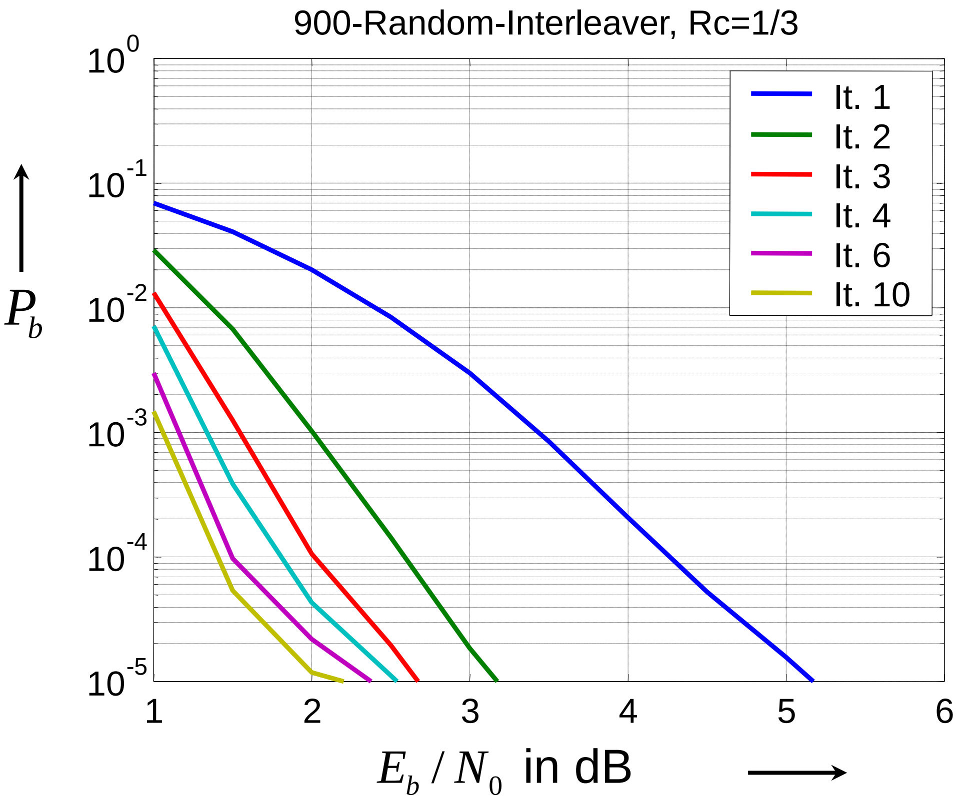


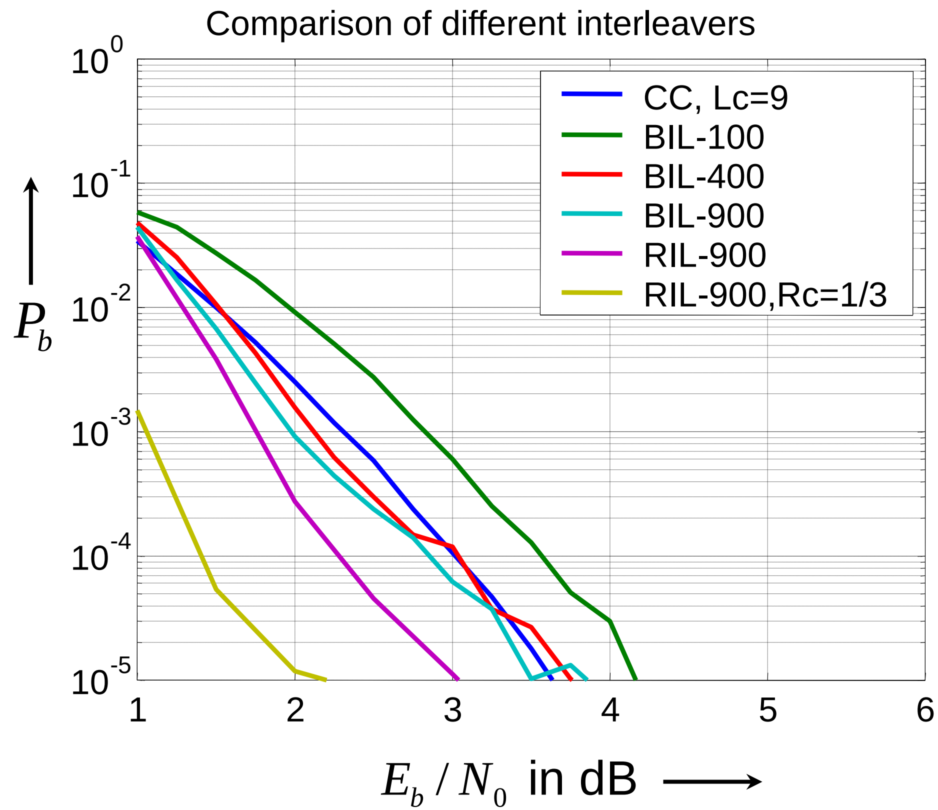
Simulation Results for Turbo Codes ($L_c = 3$)



- Gains decrease with number of iterations
- Increase of interleaver size leads to reduced BER

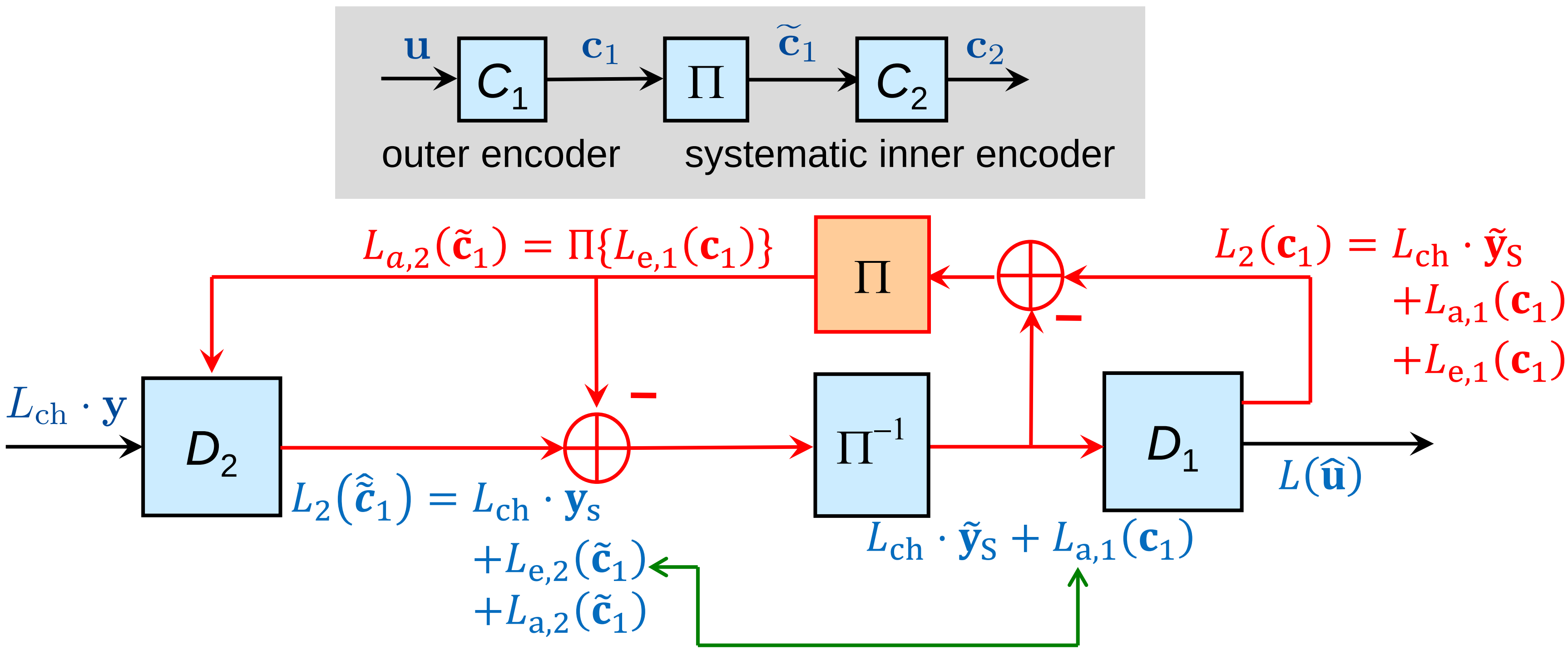
Simulation Results for Turbo Codes ($L_c = 3$)





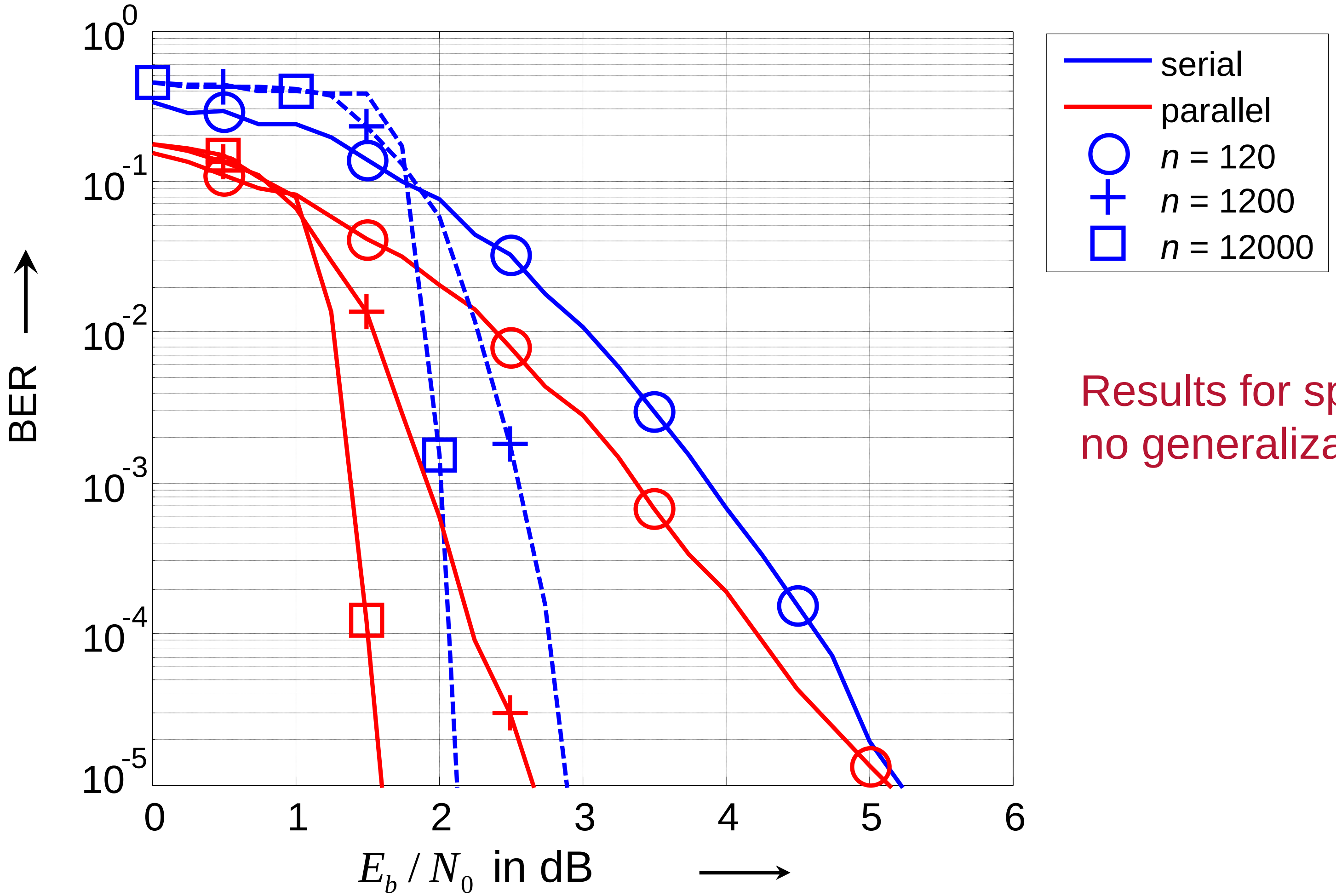
- Usage of random interleaver leads to significant performance improvements in comparison to block interleaver
- Random interleaver (RIL) achieves larger gains in comparison to block interleaver (BIL)

Turbo Decoding for Serially Concatenated Codes



- Outer decoder receives information only from inner decoder
- Outer decoder delivers estimates on information bits $\mathbf{u}$ as well as extrinsic LLRs of code bits $\mathbf{c}_1$ being information bits of inner code C_2
- Extrinsic LLRs of code bits $\mathbf{c}_1$ serve as a priori LLRs for inner code C_2

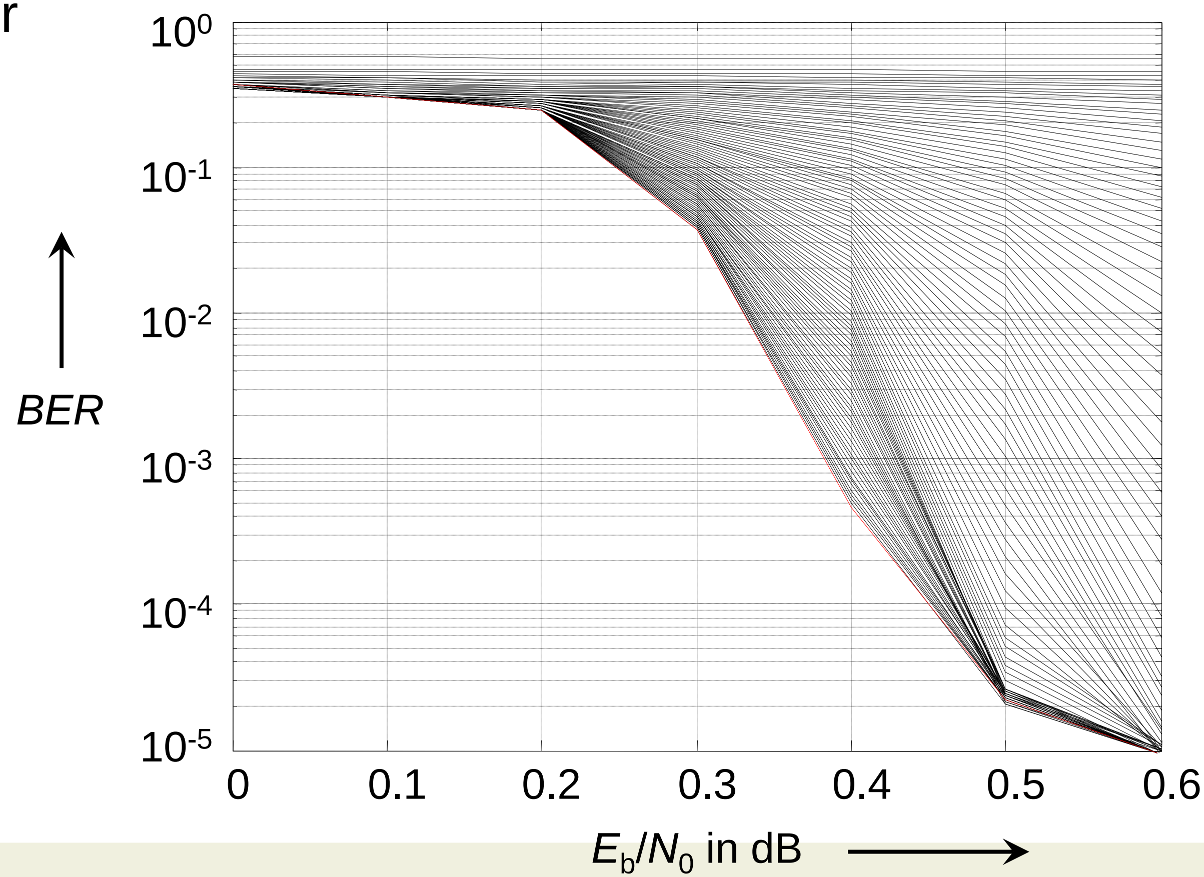
Comparison of Serial and Parallel Concatenation



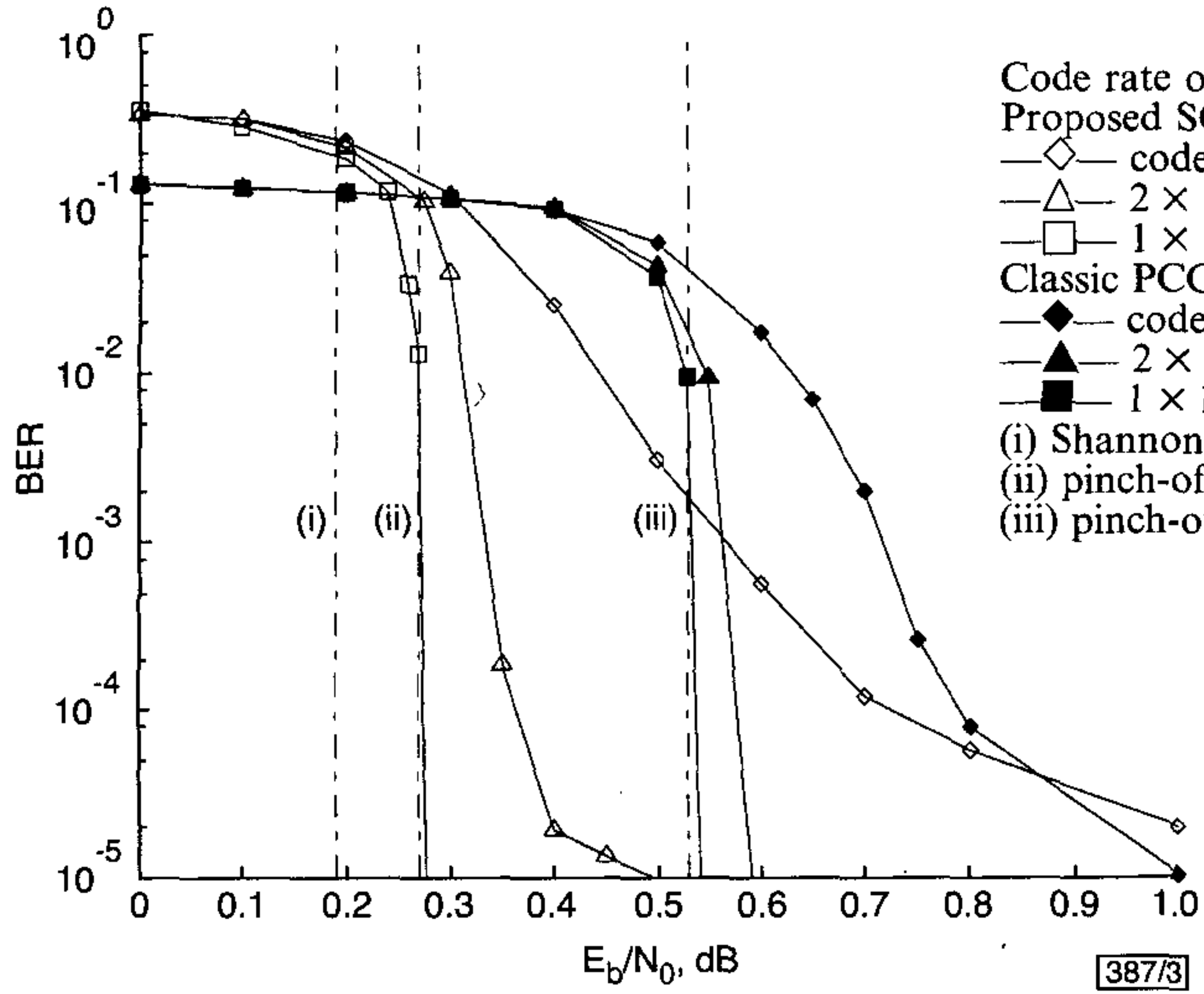
Results for specific setup, no generalization possible!

Repeat Accumulate Code by ten Brink

- Approximately 100 decoding iterations are needed
- Half-rate outer repetition encoder and rate-one inner recursive convolutional encoder



Repeat Accumulate Code by Stephan ten Brink



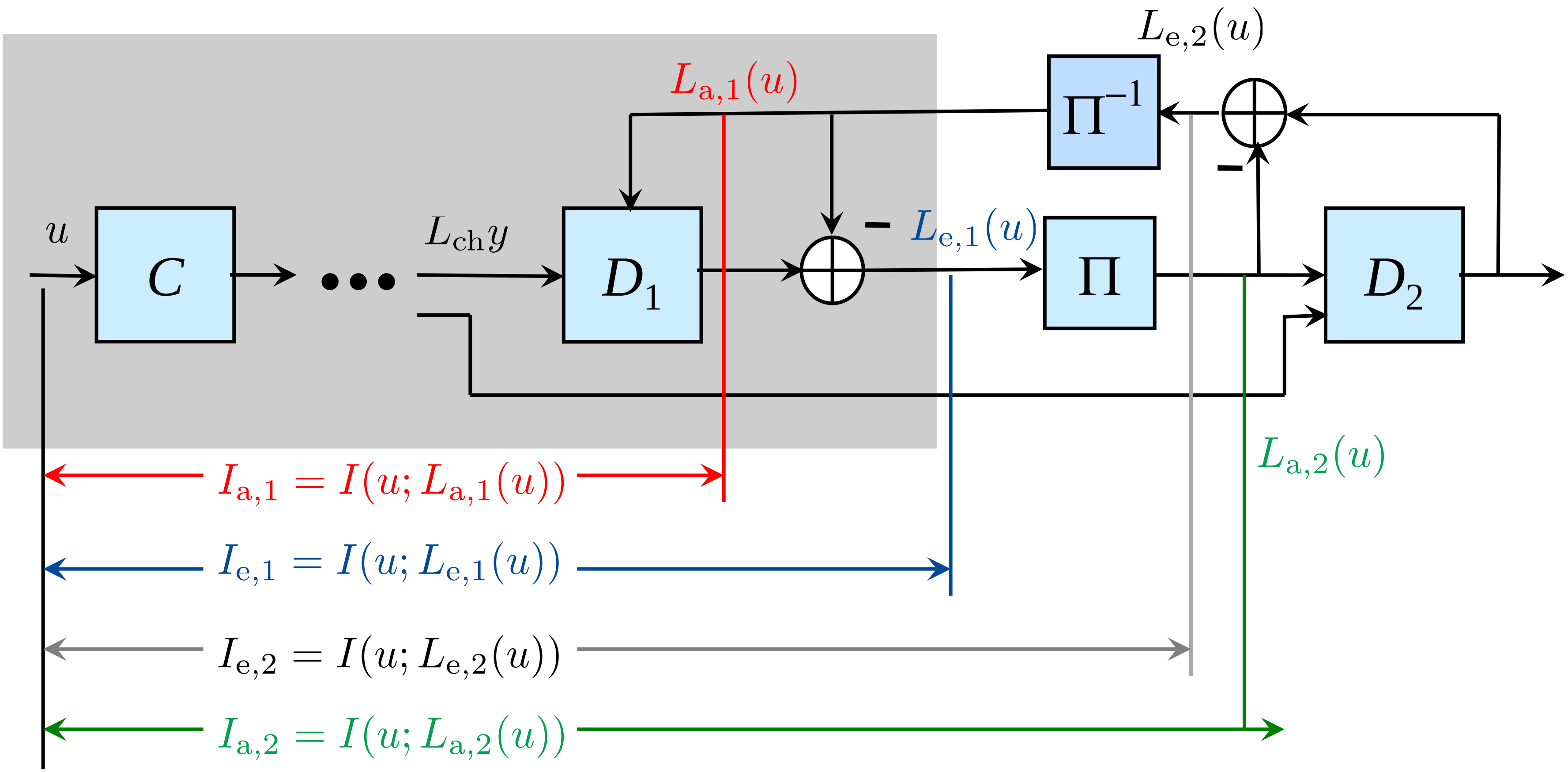
EXtrinsic Information Transfer Chart

(EXIT-Charts)

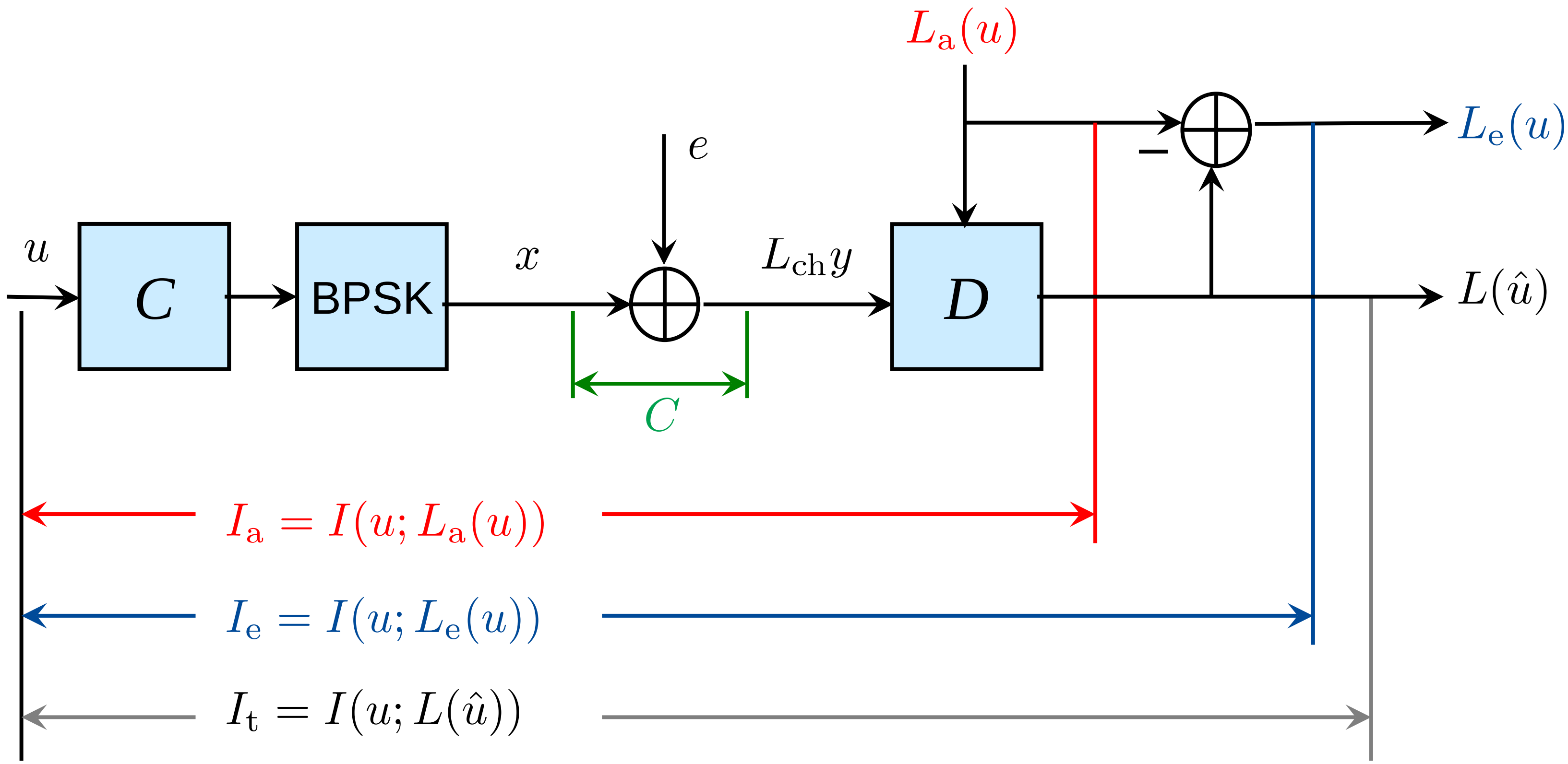
Stephan ten Brinn

Mutual Information for Turbo Decoder

- Parallel Concatenation



Mutual Information for Single Decoder



General Concept of Iterative „Turbo“ Decoding

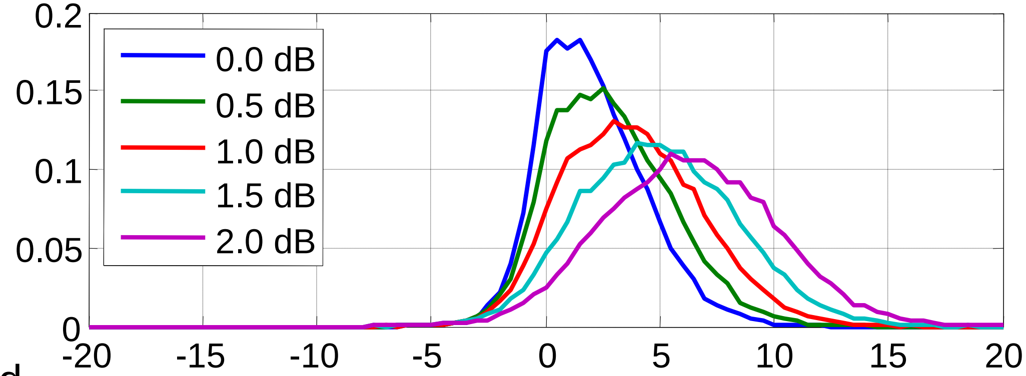
- BER curve shows three different regions
  - At low SNR the iterative decoding performs worse than uncoded transmission
  - At low to medium SNR the iterative decoding is extremely effective → waterfall region
  - At high SNR the decoding converges already in few iterations → error floor
- How to understand this varying behavior?
- Extrinsic information is exchanged between decoders
- Analysis of iterative process by semi-analytic approach
  - Determine analytically mutual information $I\big(u; L_a(u)\big)$ between information bits and a-priori input of decoder
  - Determine by simulation mutual information $I\big(u; L_e(u)\big)$ between information bits and extrinsic output of decoder for specific a-priori information at input
  - Draw relationship between both mutual information's
  - Combine diagrams of both contributing decoders into one chart:

    → **EXIT** chart: **EX**trinsic **I**nformation **T**ransfer chart

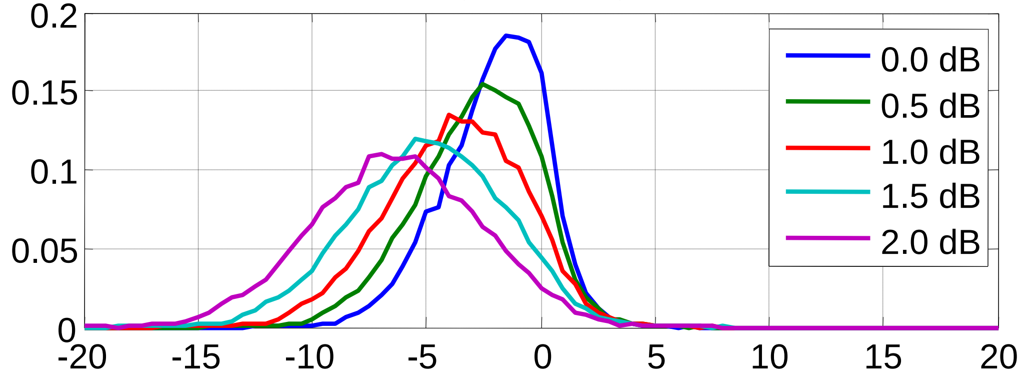
Distribution of Extrinsic Information

- Investigation of extrinsic decoder output $L_e(\hat{u}_i) = L(\hat{u}_i) - L_{ch} \cdot y_i - L_a(u_i)$
- Example: [7,5]-RSC at $E_b/N_0 = 0, \ldots, 2$ dB
  - PDF of extrinsic estimate is given for $x_i = +1$ and $x_i = -1$ separately
  - Extrinsic information is nearly Gaussian distributed
  - With increasing SNR
    - the mean's absolute value is increased
    - the variance is increased

$p_e(\xi | x_i = +1)$



$p_e(\xi | x_i = -1)$

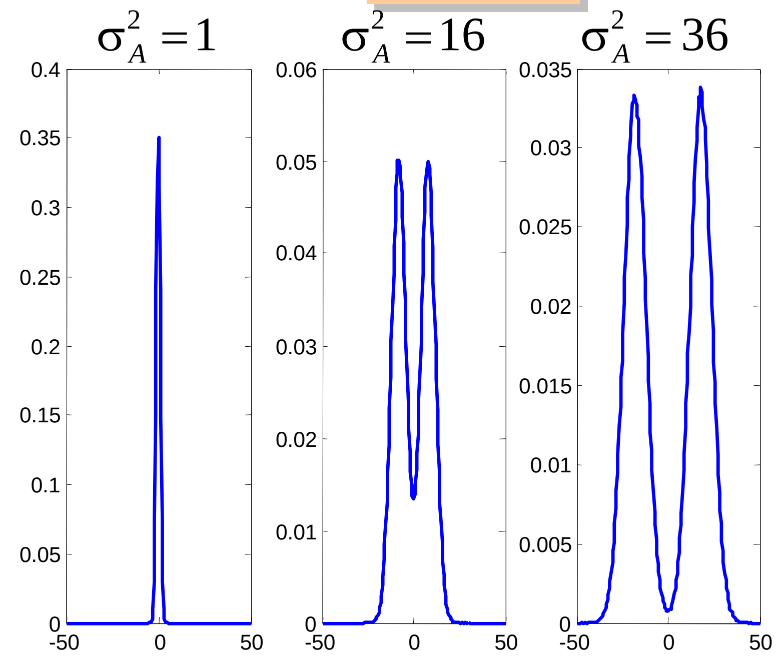


Iterative Decoding: With increasing number of iterations the extrinsic information approaches a Gaussian distribution

Analytical Model for the A-Priori Information

- Extrinsic information of decoder 1 becomes a-priori-information of decoder 2 and vice versa
- For EXIT analysis the a-priori information $A=L_a$ is modeled as $A=\mu_A \cdot x + n_A$
  - Gaussian random variable n_A of zero mean and variance σ_A^2 is added to the value x of the transmitted systematic bit multiplied by $\mu_A = \frac{1}{2}\sigma_A^2$

$$p_A\left(\xi \middle| x_i\right) = \frac{1}{\sqrt{2\pi}\sigma_A} \exp\left(- \frac{\left(\xi - \frac{\sigma_A^2}{2} \cdot x_i\right)^2}{2\sigma_A^2} \right)$$



- Normalization of a-priori information with $\frac{1}{2}\sigma_A^2$
  - With increasing variance the probability functions are more separated and do not overlap anymore

Motivation for Modeling A-Priori Information

- LLR for uncoded transmission over AWGNC is given by $y = x + n \sim \mathcal{N}\left(\pm 1, \sigma_n^2\right)$

$$L\left(y|x\right) = \ln \frac{p\{y \mid x = +1\}}{p\{y \mid x = -1\}} = \underbrace{4\frac{E_s}{N_0}}_{L_{ch}} y = L_{ch} \cdot y = L_{ch} \cdot \left(x + n\right)$$

with $L_{ch} = 4\frac{E_s}{N_0} = 4\frac{1}{2\sigma_n^2} = \frac{2}{\sigma_n^2}$

and $\sigma_n^2 = \frac{N_0}{2T_s}$ $\quad \sigma_x^2 = \frac{E_s}{T_s} = 1$

$$\Rightarrow \quad L\left(y|x\right) = \frac{2}{\sigma_n^2} \cdot x + \frac{2}{\sigma_n^2} \cdot n$$

- LLR is Gaussian distributed with mean μ_A and variance $\sigma_A{}^2$

$$\mu_A = E\left\{L\left(y|x = i\right)\right\} = \frac{2}{\sigma_n^2} \cdot i$$

$$\sigma_A^2 = E\left\{\left(\tfrac{2}{\sigma_n^2} \cdot n\right)^2\right\} = \left(\frac{2}{\sigma_n^2}\right)^2 \cdot \sigma_n^2 = \frac{4}{\sigma_n^2}$$

  - **The mean's absolute value equals the half of the variance**

- Model for a-priori LLR

$$A = L_a = \mu_A \cdot x + n_A \quad \Rightarrow \quad A \sim \mathcal{N}\left(\pm \tfrac{1}{2}\sigma_A^2, \sigma_A^2\right) = \mathcal{N}\left(\pm \frac{2}{\sigma_n^2}, \frac{4}{\sigma_n^2}\right)$$

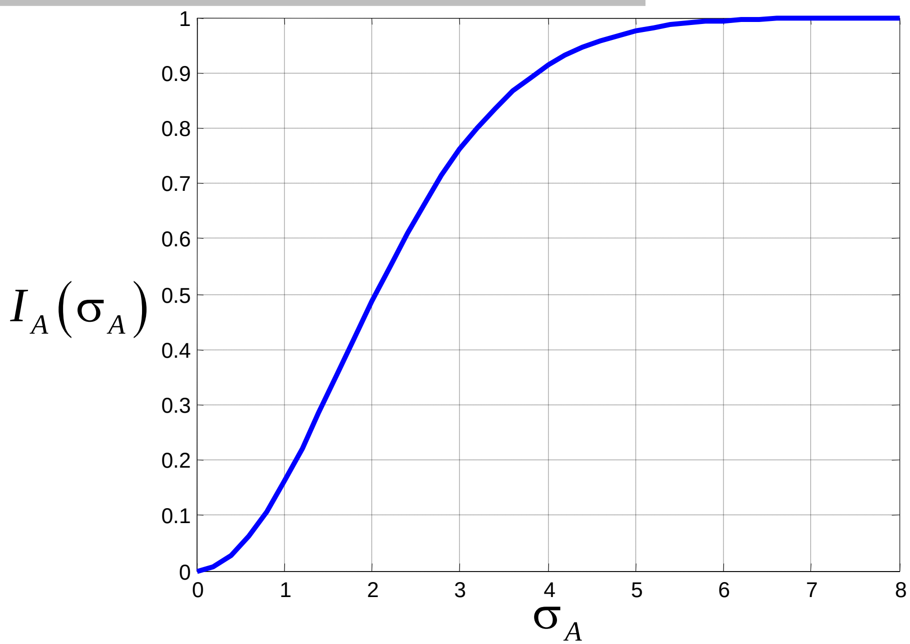
Mutual Information of A-Priori Information and Info Bits

- Mutual information between systematic bits and a-priori LLR

$$I_A(\sigma_A) = I(X;A) = \frac{1}{2} \sum_{x_i=\{+1,-1\}} \int_{-\infty}^{\infty} p_A(\xi|x_i) \log_2 \frac{2 p_A(\xi|x_i)}{p_A(\xi|x_i=-1) + p_A(\xi|x_i=+1)} d\xi$$

$$= 1 - \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma_A} \exp\left(-\frac{1}{2\sigma_A^2}\left(\xi - \tfrac{1}{2}\sigma_A^2\right)^2\right) \log_2\left(1+e^{-\xi}\right) d\xi \quad = 1 - E\left\{\log_2\left(1+e^{-\xi}\right)\right\} \quad = J(\sigma_A)$$

  - $0 \leq I_A \leq 1$
  - Integral has to be solved numerically
  - $J(\sigma_A)$ is monotonically increasing in σ_A
    → has a unique inverse function $\sigma_A = J^{-1}(I_A)$



- Close approximation for ***J*-function**

$$J(\sigma_A) = I_A(\sigma_A) \approx \left(1 - 2^{-0.3073 \cdot \sigma_A^{2 \cdot 0.8935}}\right)^{1.1064}$$

and its inverse

$$\sigma_A \approx J^{-1}(I_A) = \left(-\frac{1}{0.3073} \log 2\left(1 - I_A^{1/1.1064}\right)\right)^{\frac{1}{2 \cdot 0.8935}}$$

Mutual Information of Extrinsic Information and Info Bits

- Mutual information between systematic bits and extrinsic LLR

$$I_E = I(X;E) = \frac{1}{2} \sum_{x_i=\{+1,-1\}} \int_{-\infty}^{\infty} p_E(\xi|x_i) \log_2 \frac{2 p_E(\xi|x_i)}{p_E(\xi|x_i=-1) + p_E(\xi|x_i=+1)} d\xi$$

  - $0 \leq I_E \leq 1$

- Semi analytical approach to determine the dependency of mutual information at decoder input and output
  - Perform encoding for a random information sequence $\mathbf{u} \rightarrow \mathbf{c} = f(\mathbf{u})$ and $\mathbf{x} = 1\text{-}2\mathbf{c}$
  - Transmit BPSK signals over AWGN channel $\mathbf{y} = \mathbf{x} + \mathbf{n}$
  - For given I_A determine σ_A using the inverse J-function $\sigma_A = J^{-1}(I_A)$
  - Model a-priori information using analytical model: $\mathbf{A} = \mu_A \mathbf{x} + \mathbf{n}_A$
  - Perform decoding of noisy receive signal $\mathbf{y}$ using a-priori information $\mathbf{A}$
  - Determine mutual information I_E for extrinsic information using histogram for approximating pdf $p_E(\xi|x_i)$

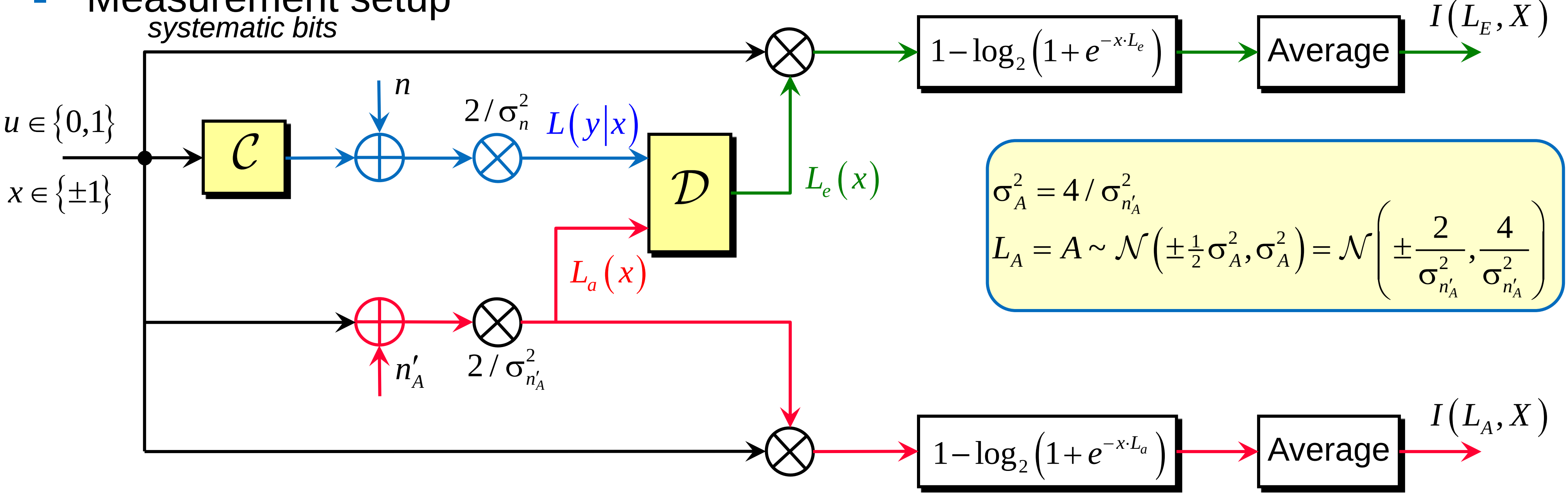
→ **Transfer characteristic** shows dependency of I_E and I_A $\quad I_E = Tr(I_A, E_b/N_0)$

Measurement of the Mutual Information

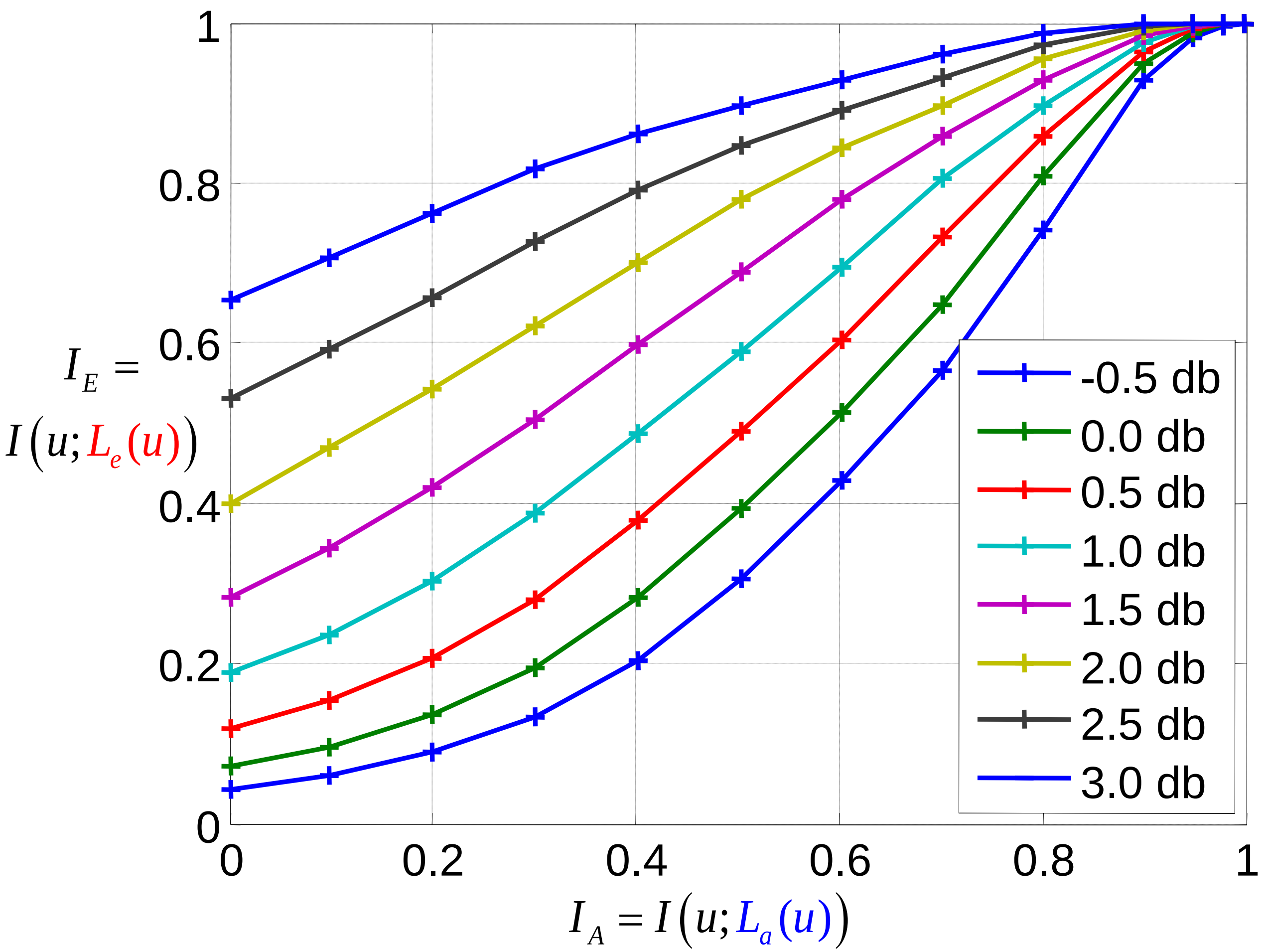
- By application of ergodic theorem (expectation is replaced by time average), the mutual information can be measured for large number N of samples

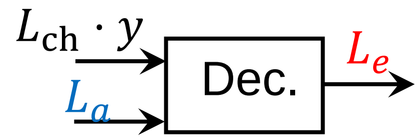
$$I(L;X) = 1 - E\left\{\log_2\left(1+e^{-L}\right)\right\} \approx 1 - \frac{1}{N}\sum_{n=1}^{N}\log_2\left(1+e^{-x_n \cdot L_n}\right)$$

- Measurement setup



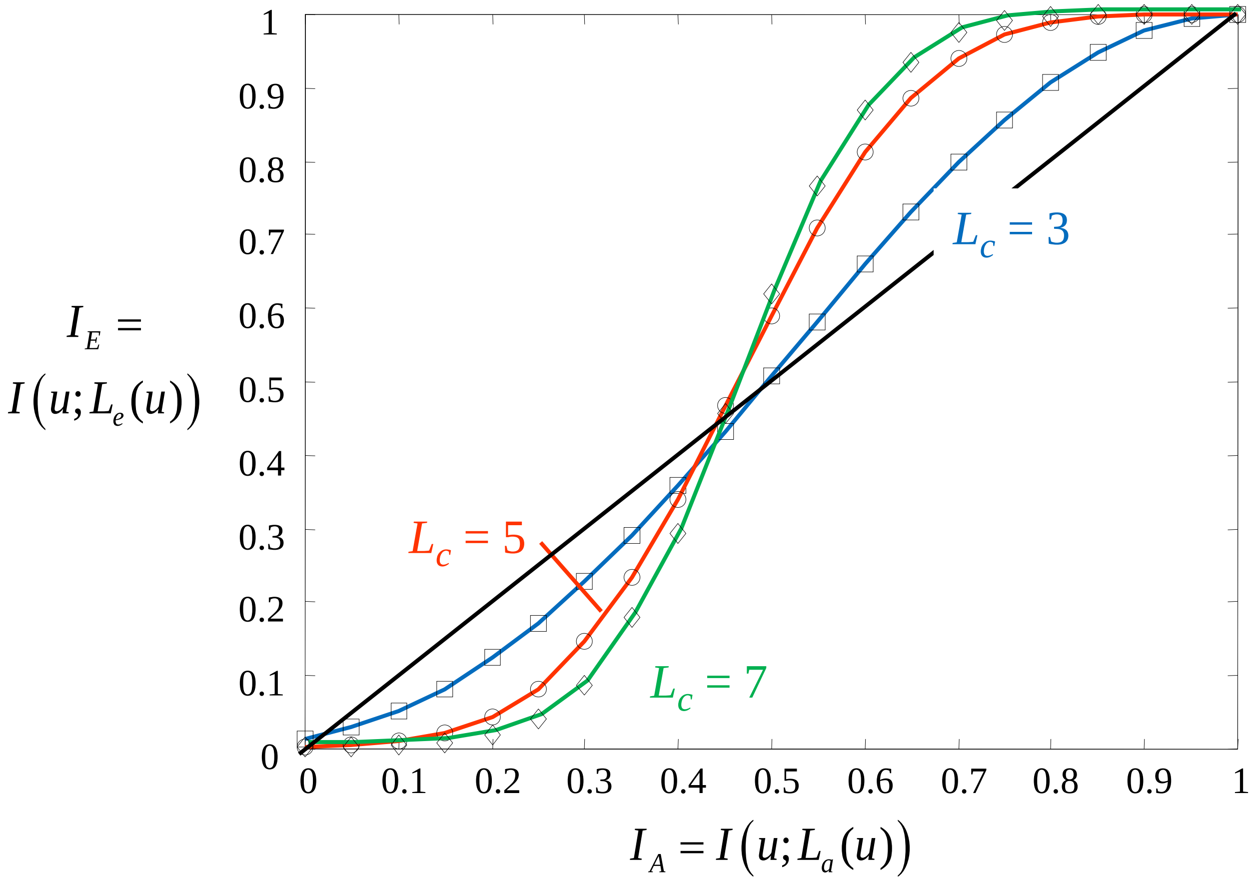
Dependency of Mutual Information at Decoder Input and Output





- Transfer characteristic for $(37,23_r)_8$- RSC code
  - Decoder processes $\mathrm{L}(y|x)$ and $\mathrm{L}_a(x)$
- Observations
  - I_E increases with growing SNR and I_A
  - I_A=0 → no a-priori information available
  - I_A=1 → perfect a-priori → I_E is reliable regardless of SNR
  - For high SNR, nearly no a-priori information is required for good decoding results

Behavior of different Convolutional Codes

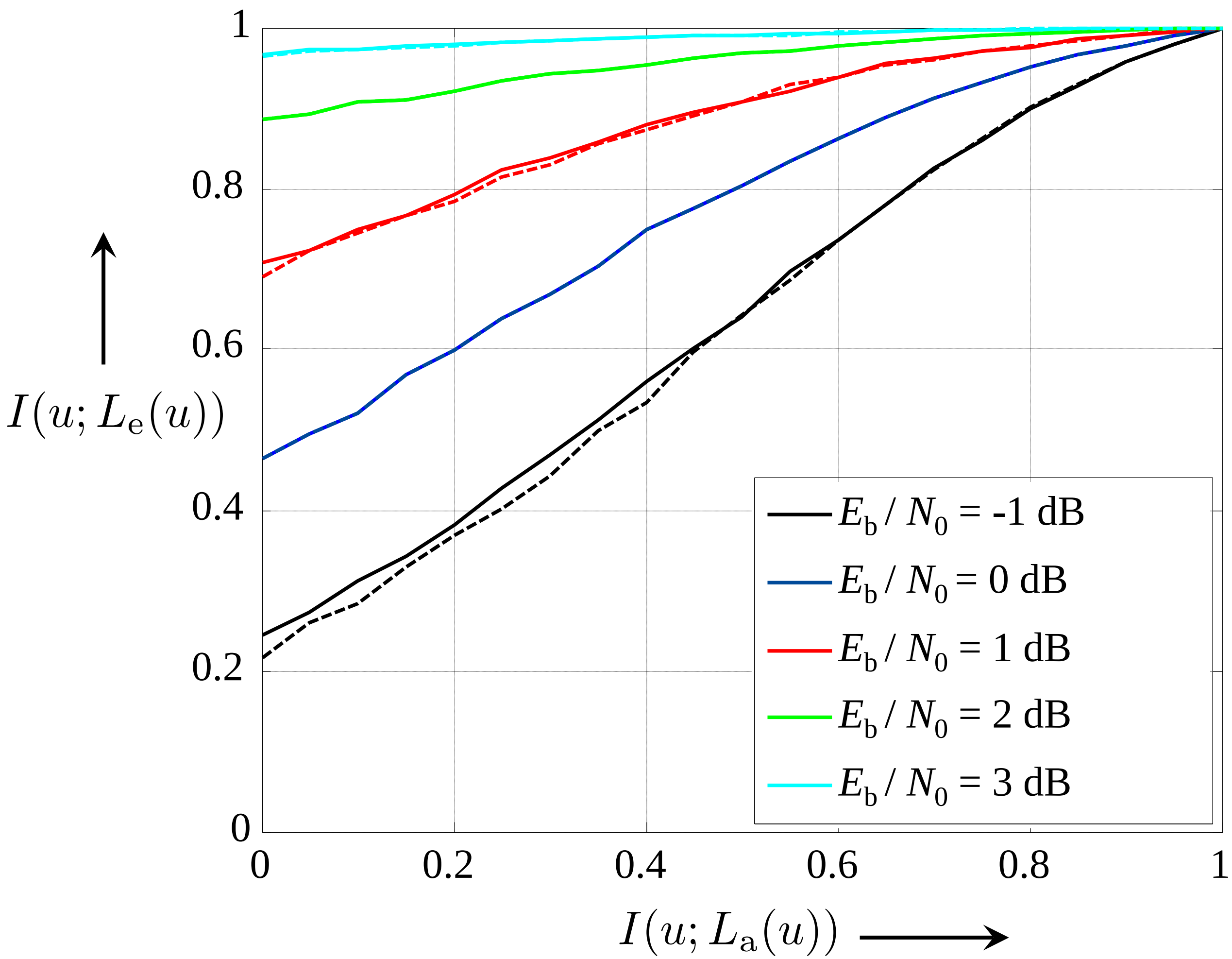


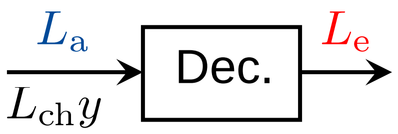
- Transfer characteristic if only a-priori information is provided to the decoder (c.f. serial concatenation)

- Weak codes better for low a-priori information
- Strong codes better for high a-priori information
- Point of intersection for all convolutional codes close to (0.5,0.5) (explanation for this behavior unknown!)

Serial concatenation: Outer decoder gets only a-priori information of inner decoder
→ Transfer function of outer decoder is independent of SNR

Comparison of MAP and Max-Log-MAP

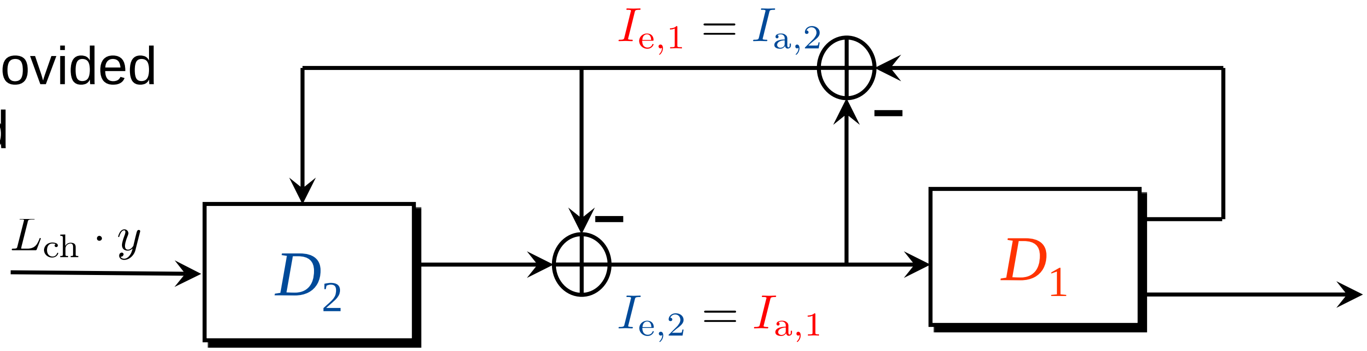




- High channel SNR leads to high extrinsic information
- Large a-priori information can compensate bad channel conditions
- Max-Log-MAP decoder performs nearly as good as optimal MAP decoder

EXtrinsic Information Transfer (EXIT) Charts

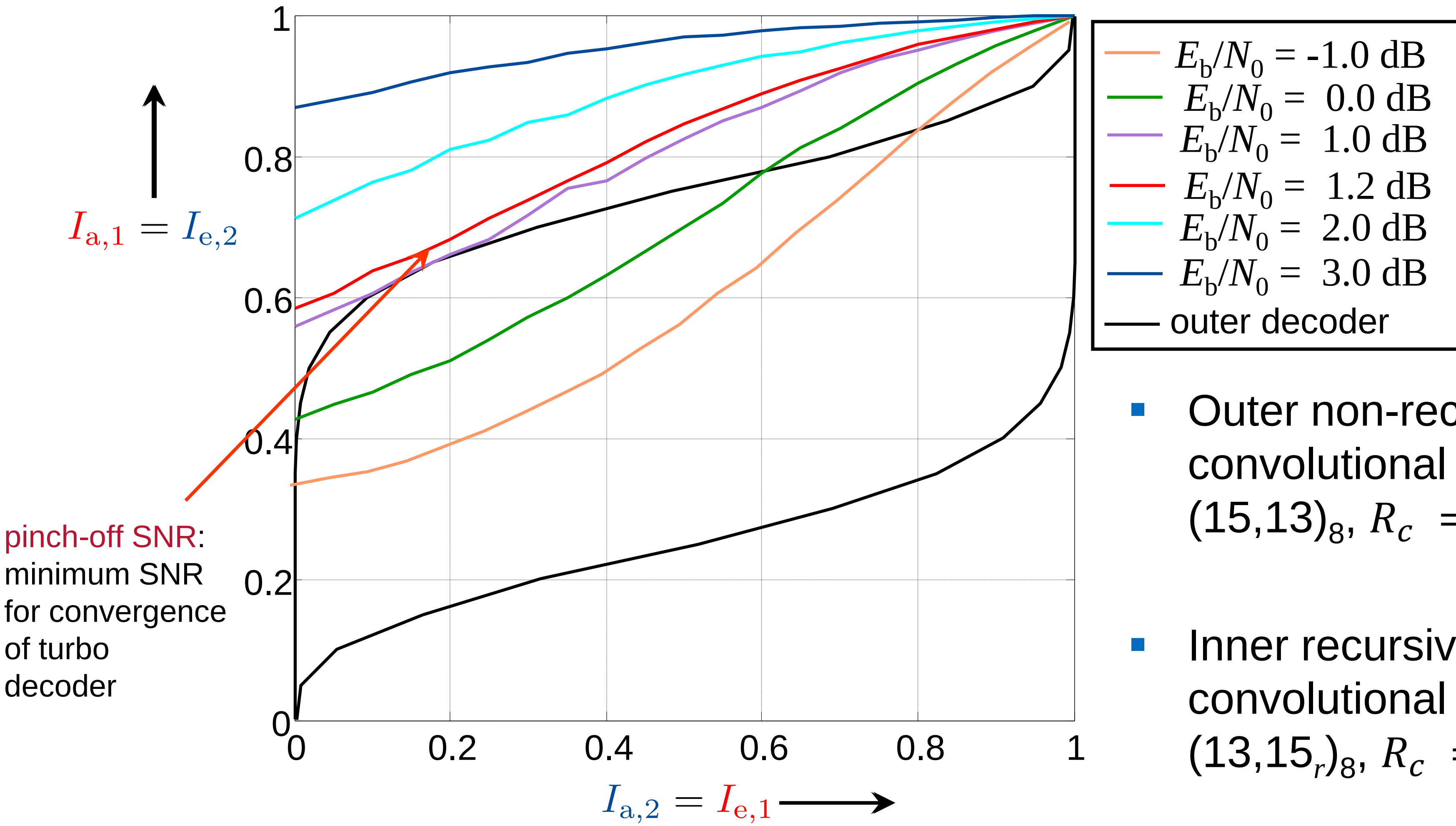
- Extrinsic information provided by one decoder is used as a-priori information for other decoder



- For EXIT charts the transfer function of both constituent codes are drawn into one diagram with exchanging the abscissa and ordinate for the second code

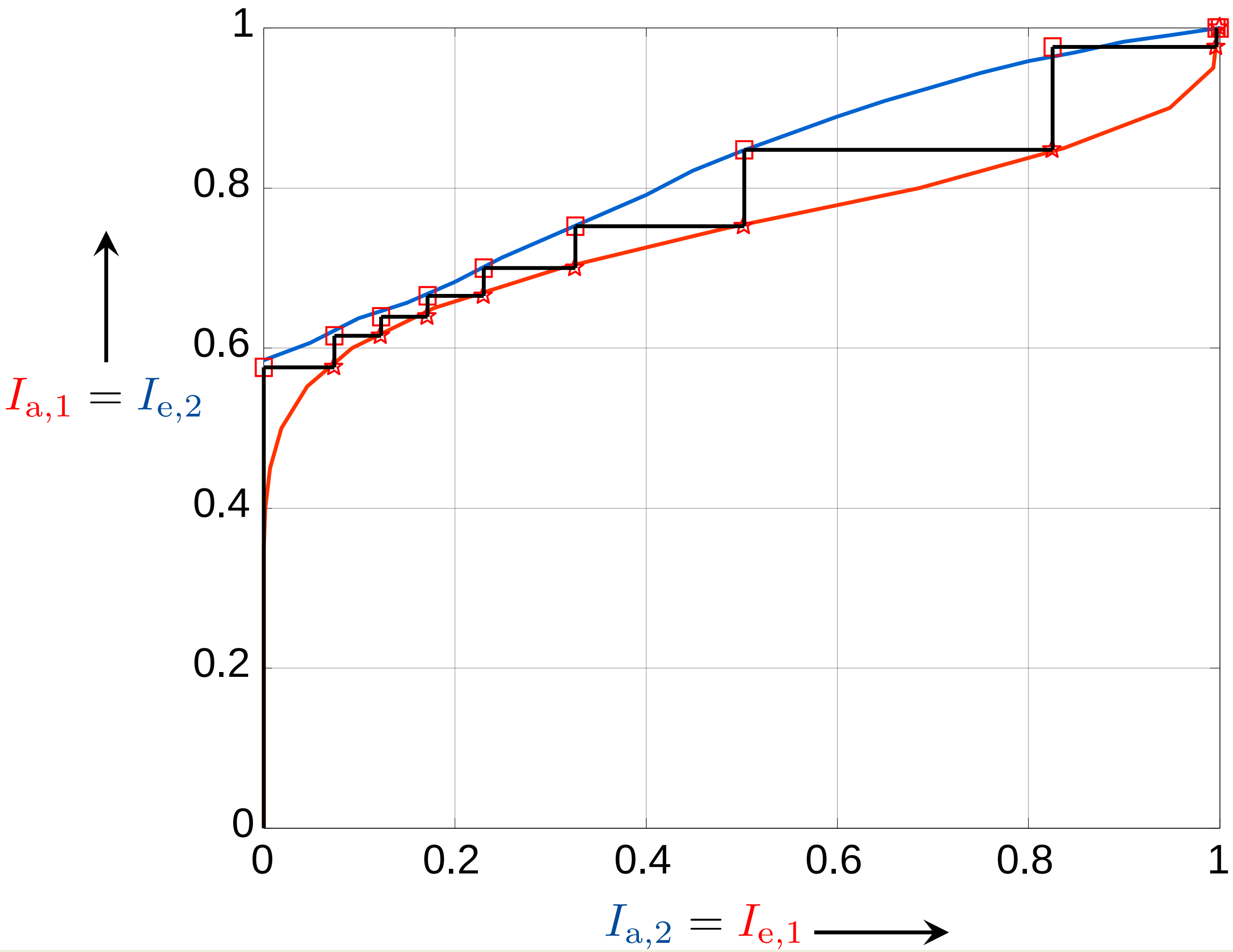
- Assumptions
  - A large interleaver is assumed to assure statistical independence of I_A and I_E
  - For inner decoders in a serial concatenated scheme and for parallel concatenated schemes the input parameters are L_{ch} and I_A
  - For outer decoders in a serial concatenation only $I_A^{(\mathrm{outer})}$ appears as input which is taken form the interleaved signal $I_E^{(\mathrm{inner})}$
    (Transfer function of outer decoder is independent of SNR)

EXIT Charts for Serial Concatenation



- Outer non-recursive convolutional encoder $(15,13)_8$, $R_c = 3/4$
- Inner recursive convolutional encoder $(13,15_r)_8$, $R_c = 2/3$

EXIT Charts for Serial Concatenation



- Outer non-recursive convolutional encoder $(15,13)_8$, $R_c = 3/4$
- Inner recursive convolutional encoder $(13,15_r)_8$, $R_c = 2/3$

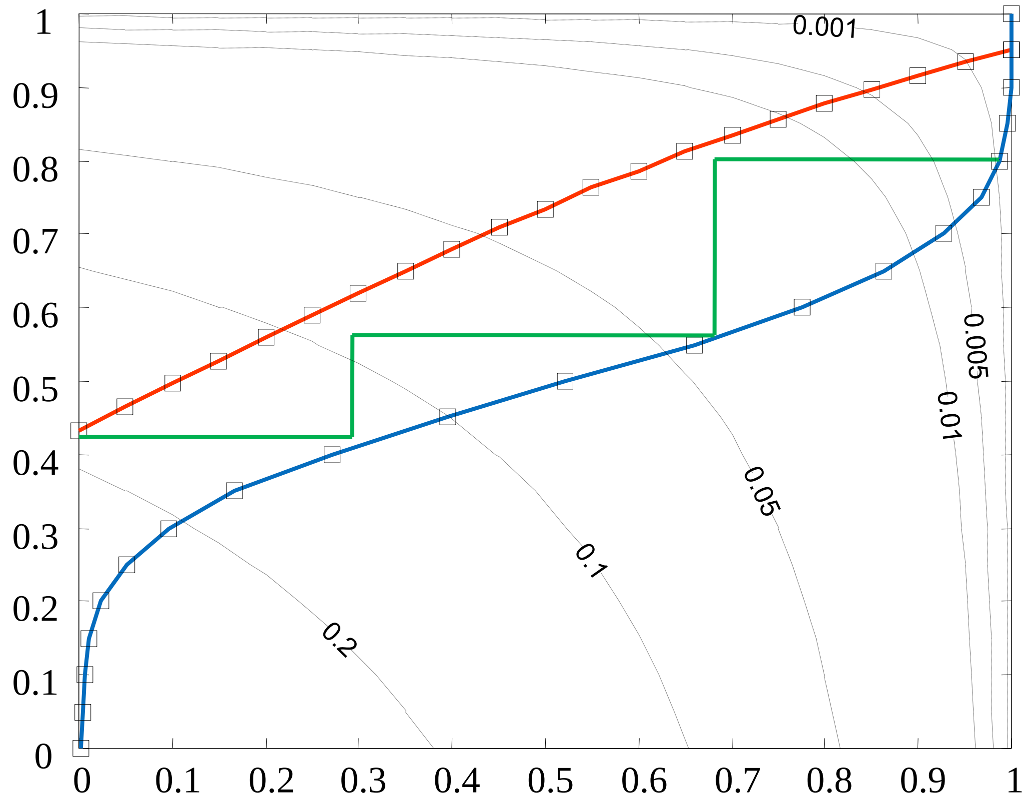
EXtrinsic Information Transfer (EXIT) Charts

$$P_b \approx \frac{1}{2}\operatorname{erfc}\left(\frac{1}{2\sqrt{2}}\sqrt{8R_c\frac{E_b}{N_0}+J^{-1}(I_A)^2+J^{-1}(I_E)^2}\right)$$

Outer convolutional code

Inner Walsh-Hadamard code

$$\bar{I}\left(u;L_e^1(u)\right)=\bar{I}\left(u;L_a^2(u)\right)$$

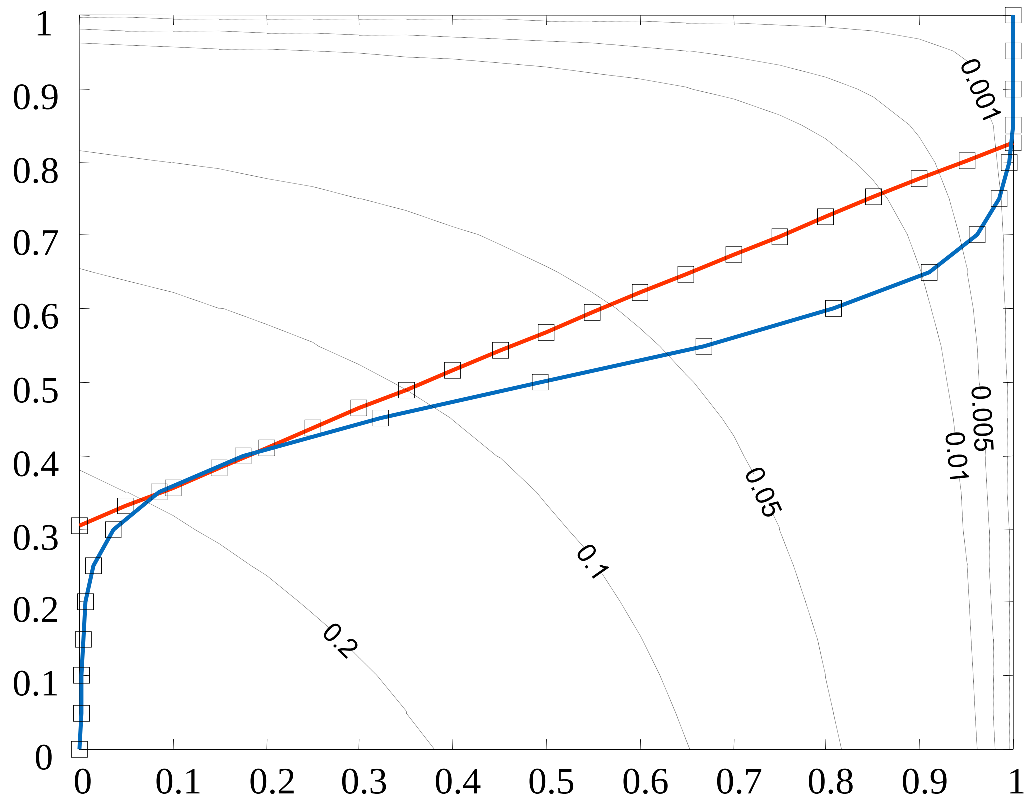


$$\bar{I}\left(u;L_a^1(u)\right)=\bar{I}\left(u;L_e^2(u)\right)$$

EXtrinsic Information Transfer (EXIT) Charts

- Determining pinch-off SNR: minimum SNR for which convergence is maintained

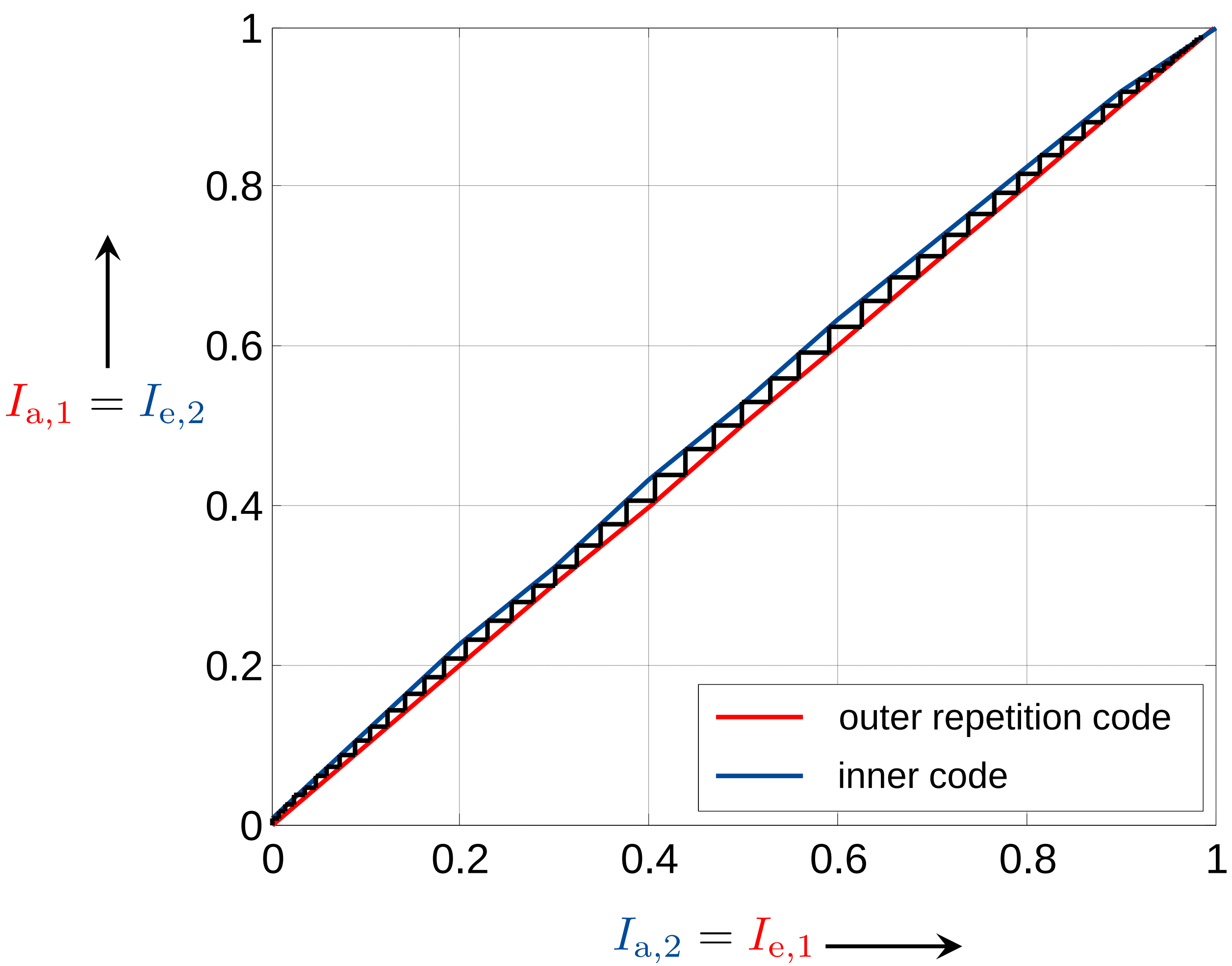
$$\bar{I}\left(u;L_e^1(u)\right)=\bar{I}\left(u;L_a^2(u)\right)$$



$$\bar{I}\left(u;L_a^1(u)\right)=\bar{I}\left(u;L_e^2(u)\right)$$

$10\log_{10}(E_b/N_0) = -0.3$ dB

Code Design for Half-Rate Repeat-Accumulate Code



Signal-to-Noise ratio

$$10\log_{10}\left(\frac{E_{\mathrm{s}}}{N_0}\right) = 0.5\ \mathrm{dB}$$

Outer repetition code

$R_{\mathrm{c}} = 1/2$

Inner recursive convolutional encoder

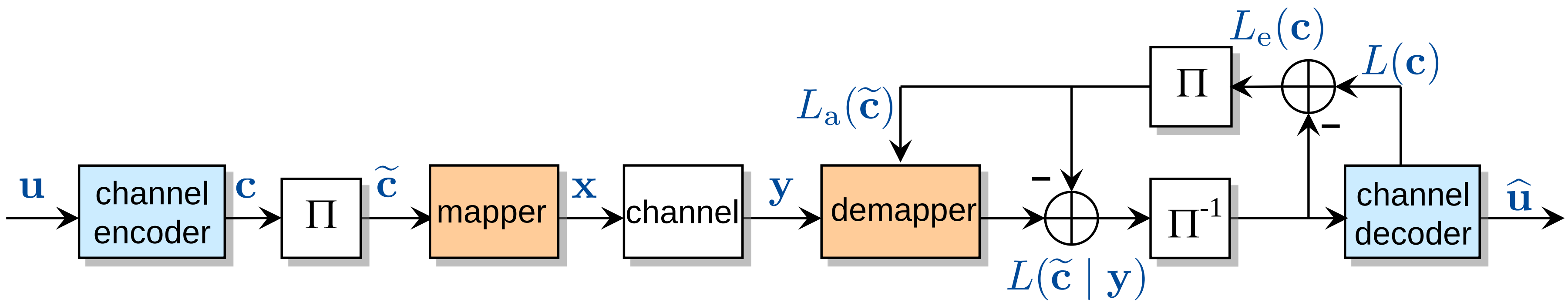
$g_1 = 1_8,\ g_2 = 7_8/15_8$

$R_{\mathrm{c}} = 1$

Bitinterleaved Coded Modulation

- General Structure for Serially Concatenated Blocks
- Calculation of LLRs
- Simulation Results

Bit-Interleaved Coded Modulation (BICM)



- Coded transmission with higher order modulation:
  - Binary vector of length m is mapped to one of 2^m symbols of the alphabet $\mathbb{X}$
  - Usually Gray mapping employed $x \in \mathbb{X}$ → minimizes bit error probability without channel coding
  - Good properties regarding the capacity of a BICM system
- Interpretation as serially concatenated system
  - Insertion of interleaver between encoder and mapper leads to pseudo random mapping of bits onto specific levels and is crucial for iterative turbo detection
- Iterative detection and decoding: demapper and decoder exchange extrinsic information

- How to perform turbo detection / decoding?
- Are there better mapping strategies than Gray mapping?

Dept. of Communications Engineering

University of Bremen

Soft-Output Demapping

- LLR for each of the m bits (for one specific time instant k):

$$L^{\text{dem}}(\tilde{c}_\mu) = L(\tilde{c}_\mu | y) = \ln \frac{p(y, \tilde{c}_\mu = 0)}{p(y, \tilde{c}_\mu = 1)} = \ln \frac{\sum\limits_{\mathbf{c} \in \mathrm{GF}(2)^m,\, c_\mu = 0} p(y|\mathbf{c}) \cdot \Pr\{\mathbf{c}\}}{\sum\limits_{\mathbf{c} \in \mathrm{GF}(2)^m,\, c_\mu = 1} p(y|\mathbf{c}) \cdot \Pr\{\mathbf{c}\}}$$

$$= \ln \frac{\sum\limits_{x \in \mathbb{X},\, c_\mu = 0} p(y|x) \cdot \Pr\{x\}}{\sum\limits_{x \in \mathbb{X},\, c_\mu = 1} p(y|x) \cdot \Pr\{x\}} = \ln \frac{\sum\limits_{x \in \mathbb{X}_\mu^0} \exp\left(-\frac{|y-x|^2}{\sigma_n^2}\right) \cdot \prod\limits_{\nu=1}^{m} \Pr\{c_\nu(x)\}}{\sum\limits_{x \in \mathbb{X}_\mu^1} \exp\left(-\frac{|y-x|^2}{\sigma_n^2}\right) \cdot \prod\limits_{\nu=1}^{m} \Pr\{c_\nu(x)\}}$$

- A priori information $L_a(\tilde{c}_\nu)$ provided by decoder

$$\prod_{\nu=1}^{m} \Pr\{c_\nu(x)\} = \prod_{\nu=1}^{m} \frac{e^{-c_\nu(x) L_a(\tilde{c}_\nu)}}{1 + e^{-L_a(\tilde{c}_\nu)}}$$

Soft-Output Demapping

- Denominator of a priori information cancels when inserted into $L^{\text{dem}}(\tilde{c}_\mu)$

$$L^{\text{dem}}\left(\tilde{c}_\mu\right) = L\left(\tilde{c}_\mu \middle| y\right) = \ln \frac{\sum_{x \in \mathbb{X}_\mu^0} \exp\left(-\frac{|y-x|^2}{\sigma_n^2}\right) \cdot \prod_{\nu=1}^{m} e^{-c_\nu(x) L_a(\tilde{c}_\nu)}}{\sum_{x \in \mathbb{X}_\mu^1} \exp\left(-\frac{|y-x|^2}{\sigma_n^2}\right) \cdot \prod_{\nu=1}^{m} e^{-c_\nu(x) L_a(\tilde{c}_\nu)}}$$

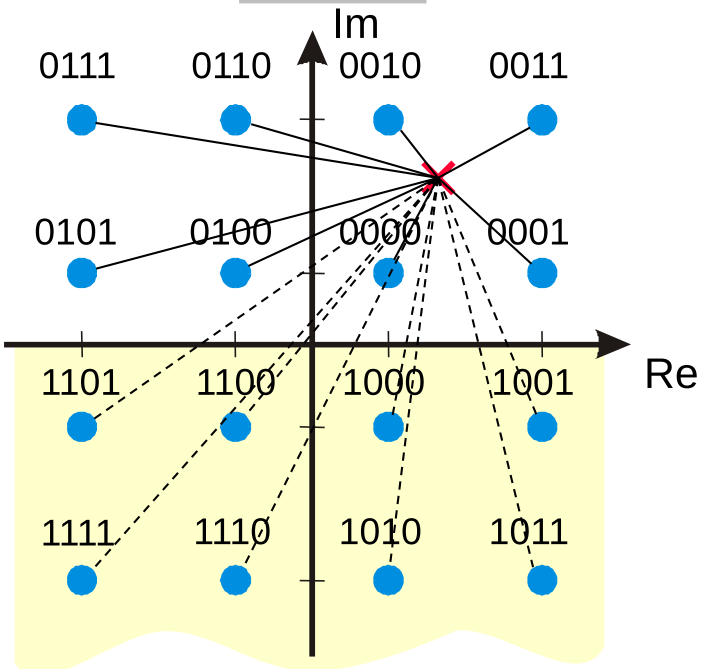
- Intrinsic information $L_i^{\text{dem}}(\tilde{c}_\nu)$ is independent of a priori information $L_a(\tilde{c}_\nu)$

$$L_i^{\text{dem}}\left(\tilde{c}_\mu\right) = L^{\text{dem}}\left(\tilde{c}_\mu\right) - L_a\left(\tilde{c}_\mu\right)$$

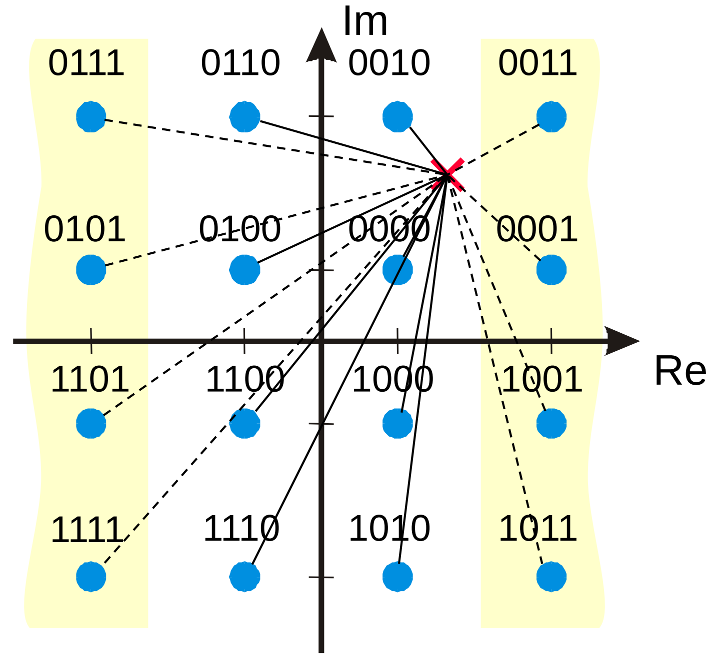
$$= \ln \frac{\sum_{x \in \mathbb{X}_\mu^0} \exp\left(-\frac{|y-x|^2}{\sigma_n^2}\right) \cdot \prod_{\nu=1, \nu\neq\mu}^{m} e^{-c_\nu(x) L_a(c_\nu)}}{\sum_{x \in \mathbb{X}_\mu^1} \exp\left(-\frac{|y-x|^2}{\sigma_n^2}\right) \cdot \prod_{\nu=1, \nu\neq\mu}^{m} e^{-c_\nu(x) L_a(c_\nu)}}$$

Soft-Output Demapping for 16-QAM

$L^{\text{dem}}(\tilde{c}_1)$

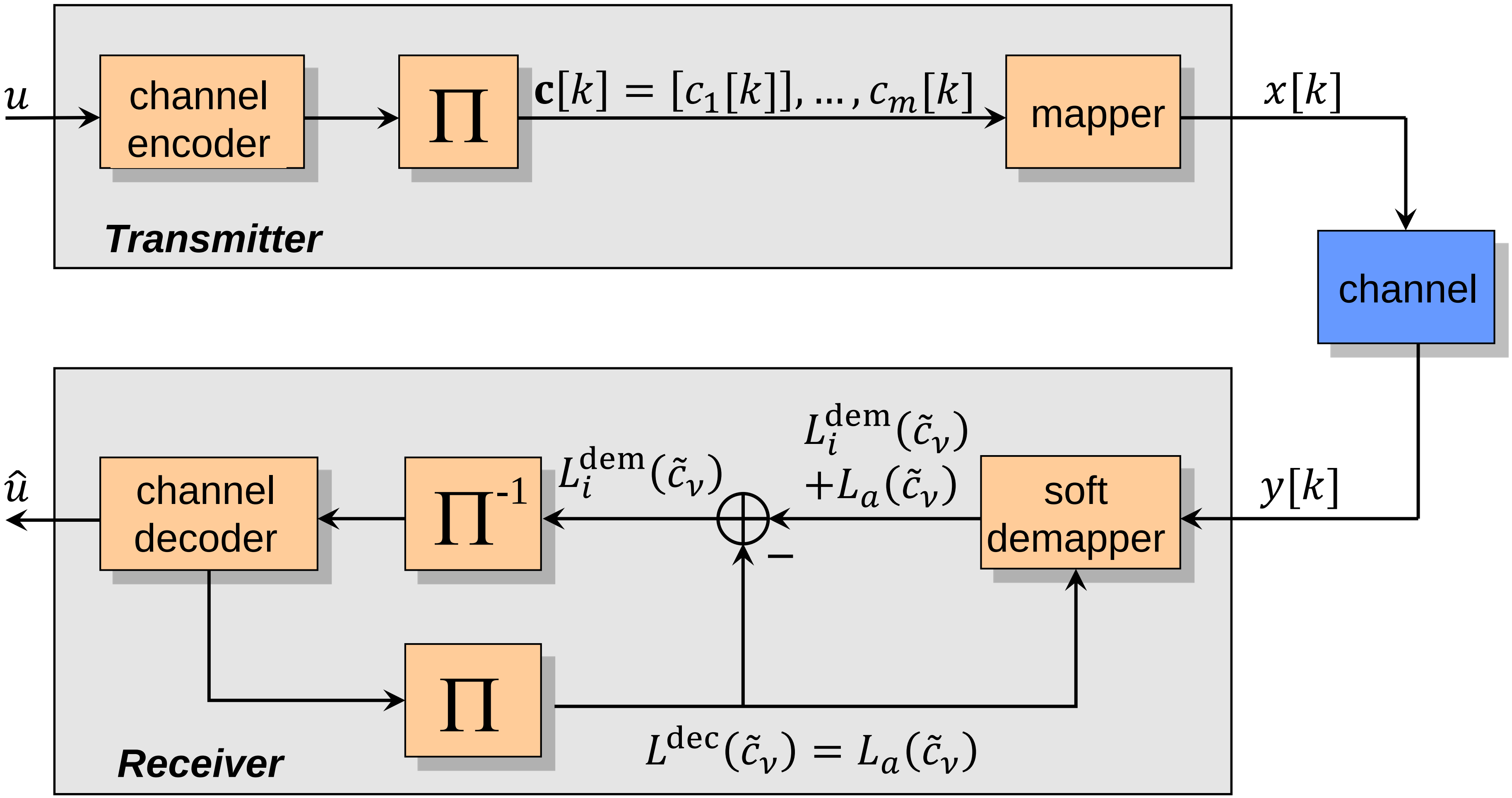


$L^{\text{dem}}(\tilde{c}_4)$

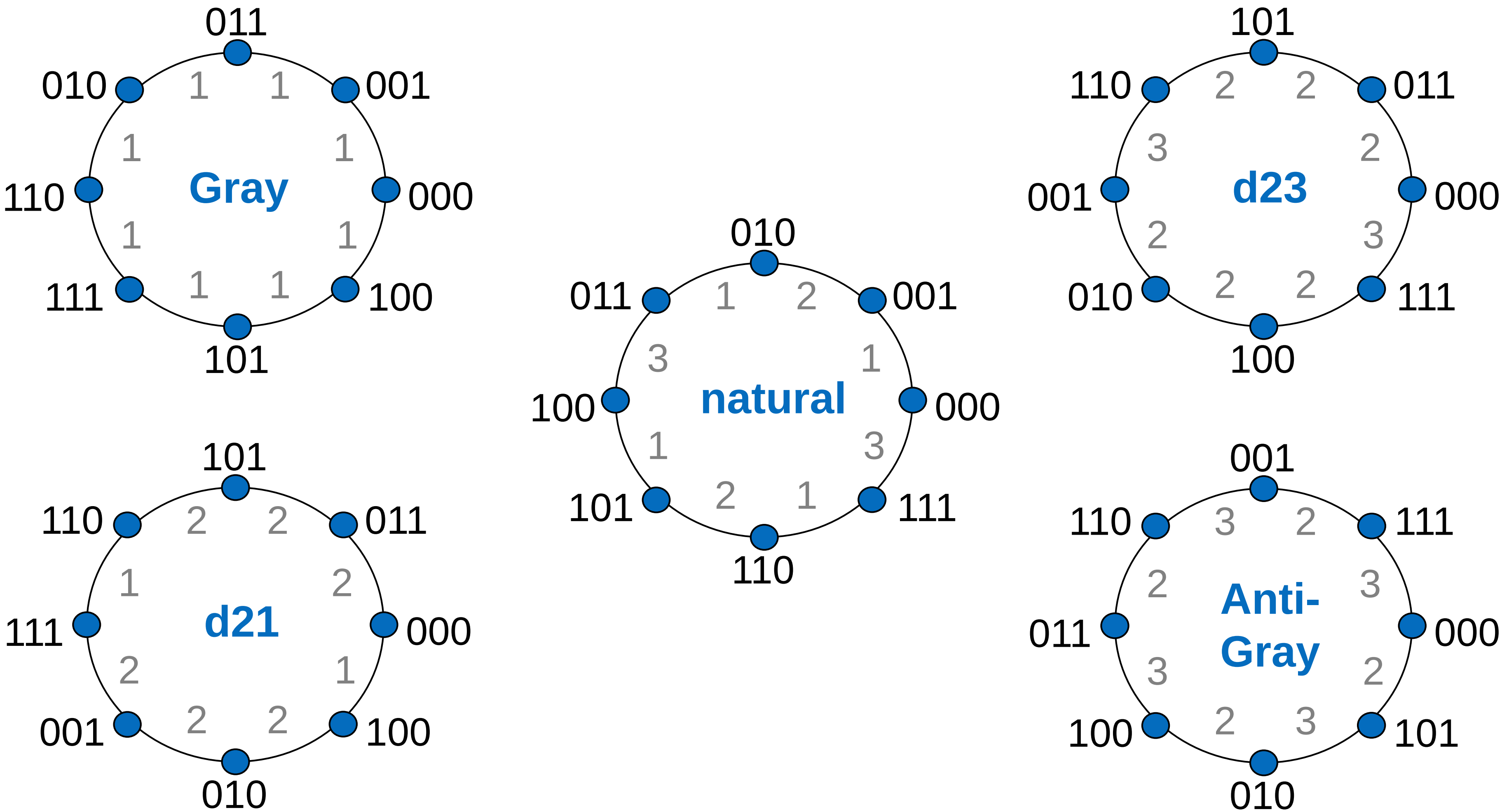


$$L^{\text{dem}}(\tilde{c}_1) = \ln \frac{\sum\limits_{x \in \mathbb{X}_1^0} \exp\left(-\frac{1}{\sigma_n^2}|y-x|^2\right) \cdot \prod\limits_{\nu=1}^{m} \Pr\{c_\nu(x)\}}{\sum\limits_{x \in \mathbb{X}_1^1} \exp\left(-\frac{1}{\sigma_n^2}|y-x|^2\right) \cdot \prod\limits_{\nu=1}^{m} \Pr\{c_\nu(x)\}}$$

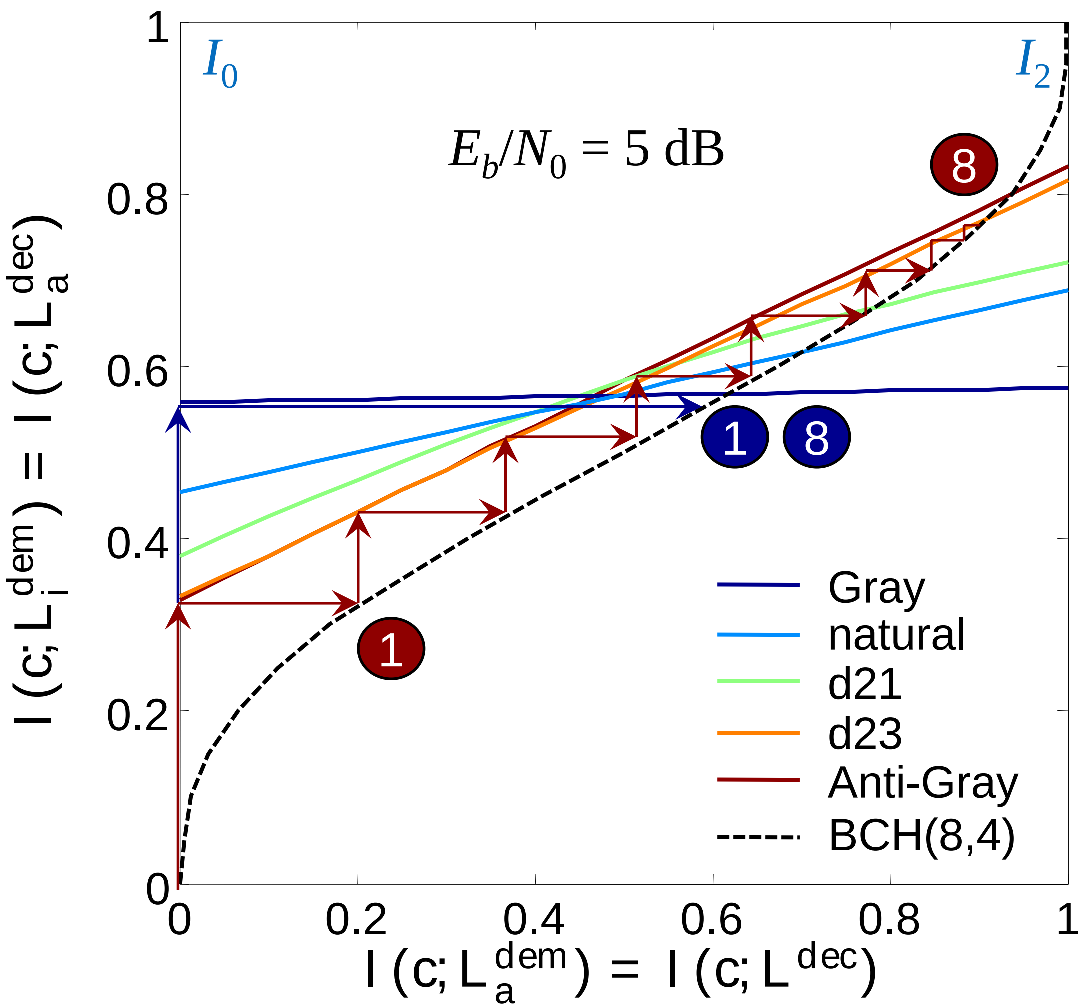
System Model for BICM



Selected Bit-Mappings for 8-PSK

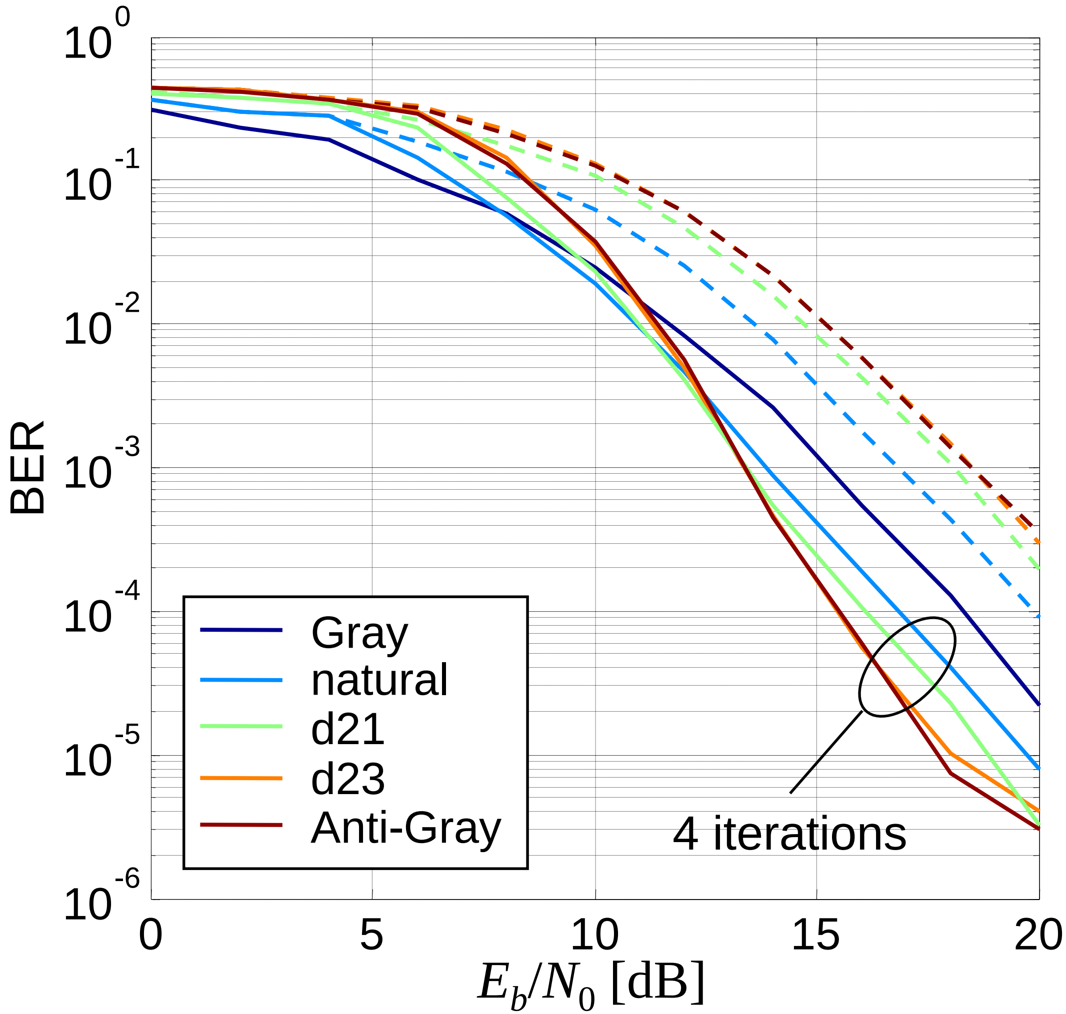


EXtrinsic Information Transfer Charts



- Demapper: *a priori* information
  → mutual information $I(c; L_a^{dem})$
- Detection and decoding only once
  - Gray is best
- Iterative detection and decoding
  - Anti-Gray is best

Bit Error Rates



- Simulation parameters
  - BCH(8,4)
  - 8-PSK
  - Alamouti scheme
  - 360 coded bits per frame
  - Independent Rayleigh fading
  - Channel const. for 24 symbols
- First detection and decoding
  - Gray good, Anti-Gray bad
- After four iterations
  - Anti-Gray is best
- Same results as predicted by EXIT charts

Low Density Parity Check Codes

- Definition and properties of LDPC codes
- Iterative decoding
- Simulation results

LDPC Codes

- **L**ow **D**ensity **P**arity **C**heck Codes
  - Invented by Robert G. Gallager in his PhD thesis, 1963
  - Re-invented by David J.C. Kay in 1999

- LDPC codes are linear block codes with sparse parity check matrix $\mathbf{H}$
  → contains relatively few ‘1’ spread among many ‘0’ (for binary codes)

- Iteratively decoded on a **factor graph** of the check matrix

- Advantages
  - Good codes
  - Low decoding complexity

Introduction

- Recall: For every linear binary (n, k) code $\mathcal{C}$ with code rate $R_c = k/n$
  - There is a **generator matrix** $\mathbf{G} \in \mathrm{GF}(q)^{k \times n}$ such that code words $\mathbf{x} \in \mathrm{GF}(q)^n$ and info words $\mathbf{u} \in \mathrm{GF}(q)^k$ are related by

$$\mathbf{x} = \mathbf{u} \cdot \mathbf{G}$$

  - There is a **parity-check** matrix $\mathbf{H} \in \mathrm{GF}(q)^{m \times n}$ of $\mathrm{rank}\{\mathbf{H}\} = n\text{-}k$, such that

$$\mathbf{x} \cdot \mathbf{H}^T = \mathbf{0}$$

  - Relation of generator and parity check matrix

$$\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0}$$

Regular LDPC-Codes

- **Definition:** A regular (d_v,d_c)-LDPC code of length n is defined by a parity-check matrix $\mathbf{H} \in \mathrm{GF}(q)^{m\times n}$ with d_v ones in each column and d_c ones in each row. The dimension of the code (info word length) is $k = n - \mathrm{rank}\{\mathbf{H}\}$

- Example:
  - $n = 8$, $m = 6$, $k = n$ - $\mathrm{rank}\{\mathbf{H}\} = 4$ (!), $R_\mathrm{C} = 1/2$
  - $d_v = 3$, $d_c = 4$

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Regular LDPC-Codes

- **Design Rate:** The true rate R_C and the design rate R_d are defined as

  $R_C = \dfrac{k}{n}$ and $R_d = 1 - \dfrac{d_v}{d_c}$ with $R_C \geq R_d$

  - Proof: The number of ones in the check matrix $m \cdot d_\mathrm{c} = n \cdot d_v$. Some parity check equations may be redundant, i.e., $m \geq n\text{-}k$, and thus $\dfrac{k}{n} = 1 - \dfrac{n-k}{n} \geq 1 - \dfrac{m}{n} = 1 - \dfrac{d_v}{d_c}$
- The check matrices can be constructed randomly or deterministic
- **Encoding**
  - LDPC codes are usually systematically encoded, i.e., by a systematic generator matrix $\mathbf{G} = \left[\mathbf{I}_{k \times k} \middle| \mathbf{P}_{k \times n-k} \right]$
  - The matrix $\mathbf{P}$ can be found by transforming $\mathbf{H}$ into another check matrix of the code, that has the form

    $$\mathbf{H}' = \left[-\mathbf{P}^T_{k \times n-k} \middle| \mathbf{I}_{n-k \times n-k} \right]$$

Factor Graph

- A **factor graph** of a code is a graphical representation of the code constraints defined by a parity-check matrix of this code

$$\mathbf{x} \cdot \mathbf{H}^T = \mathbf{0}$$

- The factor graph is a bipartite graph with
  - a **variable node** for each code symbol,
  - a **check node** for each check equation,
  - an **edge** between a variable node and a check node if the code symbol participates in the check equation

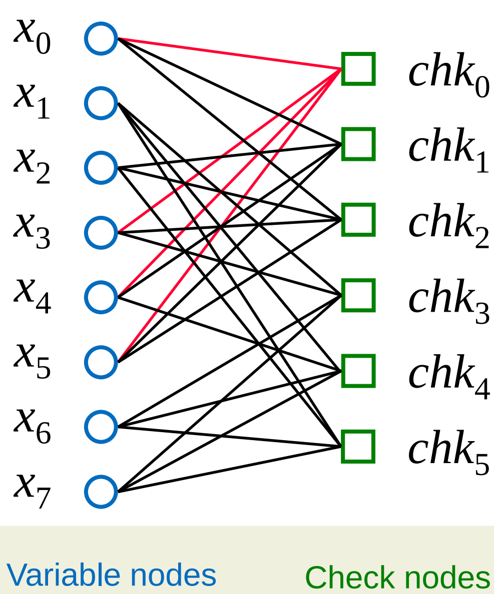
- Notice that each edge corresponds to one ‘1’ in the check matrix.

Factor Graph

- Example:

$$\mathbf{x}\cdot\mathbf{H}^T=\begin{bmatrix}x_0 & x_1 & \dots & x_7\end{bmatrix}\begin{bmatrix}1&0&0&1&1&1&0&0\\1&0&1&0&1&1&0&0\\1&0&1&1&0&1&0&0\\0&1&0&1&0&0&1&1\\0&1&0&0&1&0&1&1\\0&1&1&0&0&0&1&1\end{bmatrix}^T=\mathbf{0}$$

$x_0\oplus x_3\oplus x_4\oplus x_5=0$	chk_0
$x_0\oplus x_2\oplus x_4\oplus x_5=0$	chk_1
$x_0\oplus x_2\oplus x_3\oplus x_5=0$	chk_2
$x_1\oplus x_3\oplus x_6\oplus x_7=0$	chk_3
$x_1\oplus x_4\oplus x_6\oplus x_7=0$	chk_4
$x_1\oplus x_2\oplus x_6\oplus x_7=0$	chk_5



$n=8$ columns (code word length)

$n\text{-}k=6$ parity check equations

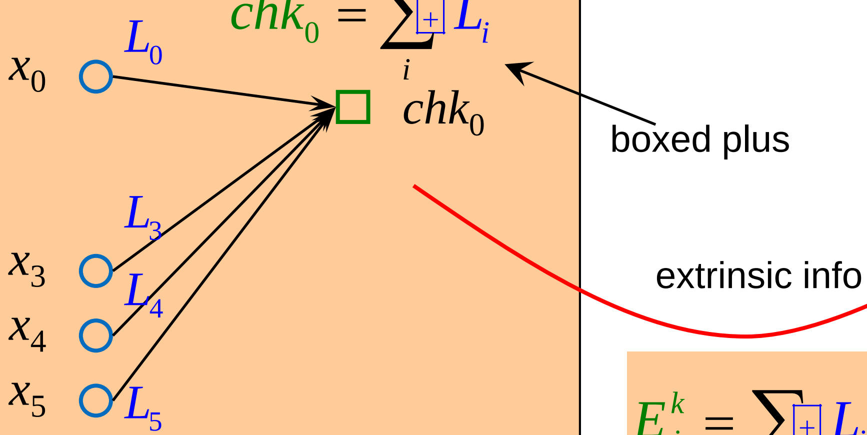
Each check node represents one row of parity check matrix

Decoding with the Sum-Product Algorithm

- Similar to Turbo Decoding, extrinsic information is exchanged
  - Check nodes „collect“ extrinsic information from the connected variable nodes
  - Variable nodes „collect“ extrinsic information from the connected check nodes

Check node example

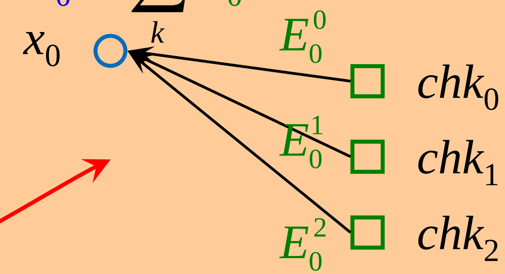
$$chk_0 = \sum_i \boxplus L_i$$



$$E_j^k = \sum_{\substack{i \neq j \\ i \in K_k}} \boxplus L_i$$

Variable node example

$$A_0 = \sum_k E_0^k$$



- Iterative decoding procedure

→ Also called „message passing“ or “believe propagation”

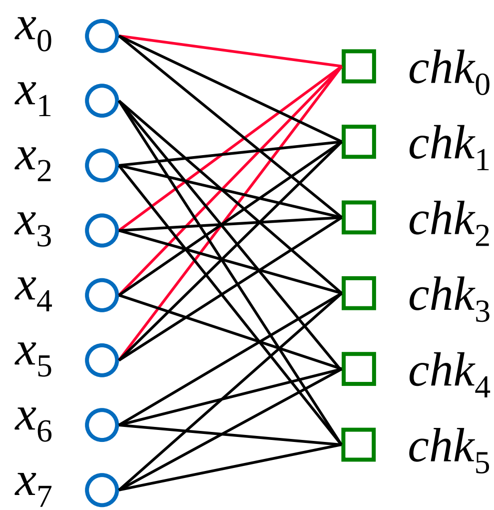
Stop if $\mathbf{x} \cdot \mathbf{H}^T = \mathbf{0}$

Decoding with the Sum-Product Algorithm

- First check equation $x_0 \oplus x_3 \oplus x_4 \oplus x_5 = 0$
  - Is the check equation fulfilled? $chk_0 = L(x_0) \boxplus L(x_3) \boxplus L(x_4) \boxplus L(x_5)$
  - Extrinsic information

$$x_0 = x_3 \oplus x_4 \oplus x_5 \;\rightarrow\; L_e^0(x_0) = L(x_3) \boxplus L(x_4) \boxplus L(x_5)$$

$$L(x_0) = L_{\mathrm{ch}}\, y_0$$
$$L(x_1) = L_{\mathrm{ch}}\, y_1$$
$$L(x_2) = L_{\mathrm{ch}}\, y_2$$
$$L(x_3) = L_{\mathrm{ch}}\, y_3$$
$$L(x_4) = L_{\mathrm{ch}}\, y_4$$
$$L(x_5) = L_{\mathrm{ch}}\, y_5$$
$$L(x_6) = L_{\mathrm{ch}}\, y_6$$
$$L(x_7) = L_{\mathrm{ch}}\, y_7$$



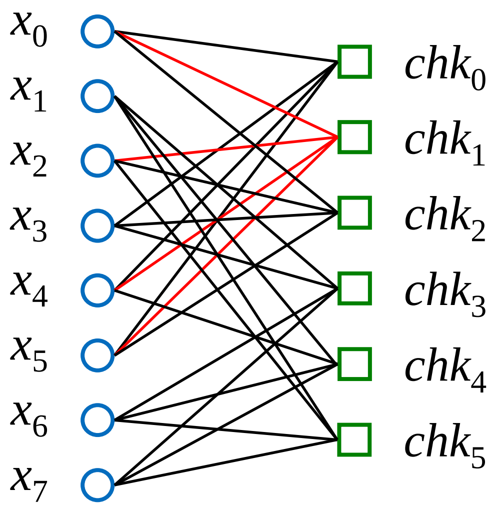
$$L_e^0(x_3) = L(x_0) \boxplus L(x_4) \boxplus L(x_5)$$
$$L_e^0(x_4) = L(x_0) \boxplus L(x_3) \boxplus L(x_5)$$
$$L_e^0(x_5) = L(x_0) \boxplus L(x_3) \boxplus L(x_4)$$

Decoding with the Sum-Product Algorithm

- Second check equation $x_0 \oplus x_2 \oplus x_4 \oplus x_5 = 0$
- Third check equation $x_0 \oplus x_2 \oplus x_3 \oplus x_5 = 0$

$$L_e^1(x_0) = L(x_2) \boxplus L(x_4) \boxplus L(x_5)$$
$$L_e^1(x_2) = L(x_0) \boxplus L(x_4) \boxplus L(x_5)$$
$$L_e^1(x_4) = L(x_0) \boxplus L(x_2) \boxplus L(x_5)$$
$$L_e^1(x_5) = L(x_0) \boxplus L(x_2) \boxplus L(x_4)$$

$$L_e^2(x_0) = L(x_2) \boxplus L(x_3) \boxplus L(x_5)$$
$$L_e^2(x_2) = L(x_0) \boxplus L(x_3) \boxplus L(x_5)$$
$$L_e^2(x_3) = L(x_0) \boxplus L(x_2) \boxplus L(x_5)$$
$$L_e^2(x_5) = L(x_0) \boxplus L(x_2) \boxplus L(x_3)$$



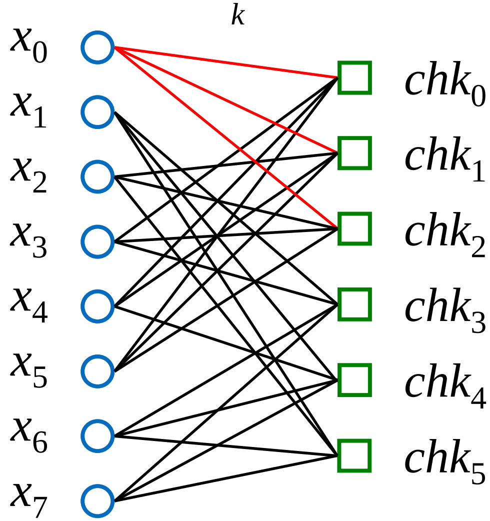
- …

Decoding with the Sum-Product Algorithm

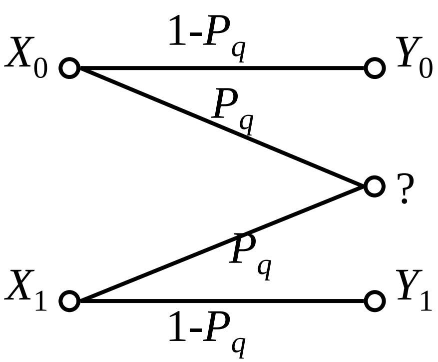
- Variable update
  - Collect extrinsic information of check nodes and update variable nodes

$$A_0 = \sum_k E_0^k$$

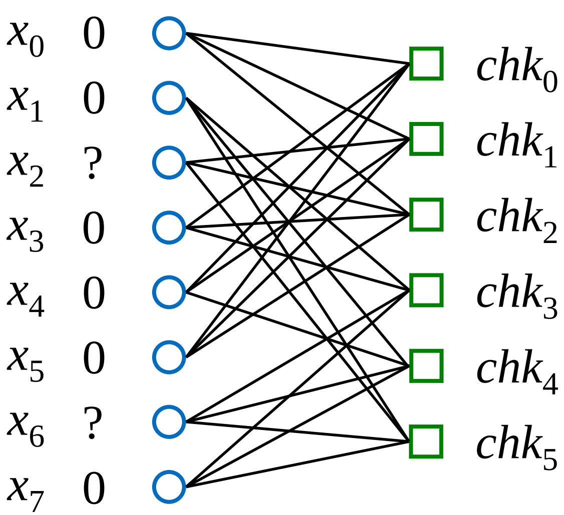
$$
\begin{aligned}
L(x_0) &= L_{\mathrm{ch}}\, y_0 + A_0 \\
L(x_1) &= L_{\mathrm{ch}}\, y_1 + A_1 \\
L(x_2) &= L_{\mathrm{ch}}\, y_2 + A_2 \\
L(x_3) &= L_{\mathrm{ch}}\, y_3 + A_3 \\
L(x_4) &= L_{\mathrm{ch}}\, y_4 + A_4 \\
L(x_5) &= L_{\mathrm{ch}}\, y_5 + A_5 \\
L(x_6) &= L_{\mathrm{ch}}\, y_6 + A_6 \\
L(x_7) &= L_{\mathrm{ch}}\, y_7 + A_0
\end{aligned}
$$



Example: BEC

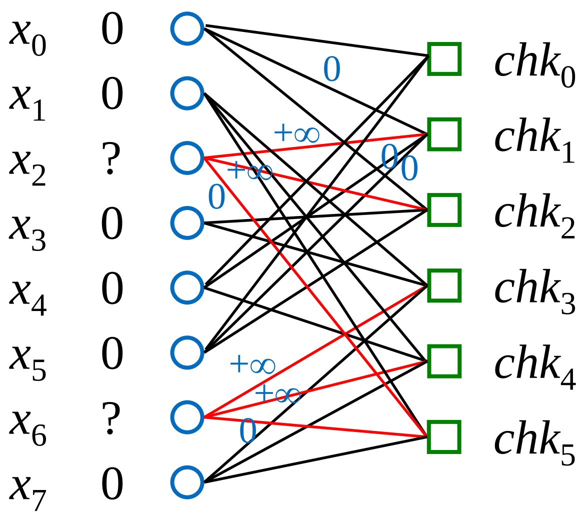


$$L(y)=\begin{cases} +\infty & y=Y_0 \\ 0 & y=? \\ -\infty & y=Y_1 \end{cases}$$



Example: BEC

- Check equations → calculate extrinsic information



$$L_e^1(x_2) = L(x_0) \boxplus L(x_4) \boxplus L(x_5) = +\infty \qquad L_e^1(x_0) = L_e^1(x_4) = L_e^1(x_5) = 0$$

$$L_e^2(x_2) = L(x_0) \boxplus L(x_3) \boxplus L(x_5) = +\infty$$

$$L_e^5(x_2) = L(x_1) \boxplus L(x_6) \boxplus L(x_7) = 0$$

$$L_e^3(x_6) = L(x_1) \boxplus L(x_3) \boxplus L(x_7) = +\infty$$

$$L_e^4(x_6) = L(x_1) \boxplus L(x_4) \boxplus L(x_7) = +\infty$$

$$L_e^5(x_6) = L(x_1) \boxplus L(x_2) \boxplus L(x_7) = 0$$

- Variable check

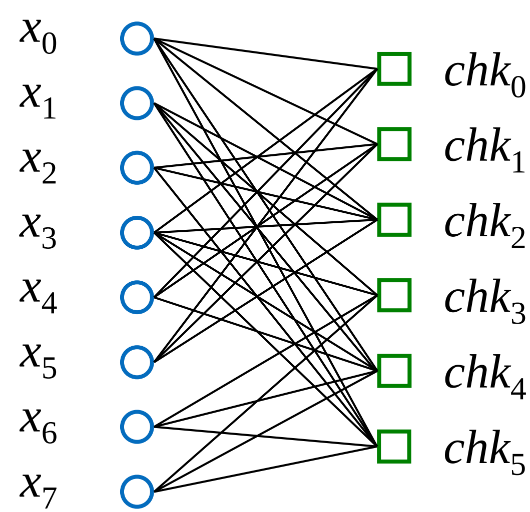
$$L_a(x_2) = L_e^1(x_2) + L_e^2(x_2) + L_e^5(x_2) = 0 \qquad L_e^5(x_2) = L_e^1(x_2) + L_e^2(x_2) = +\infty$$

$$L_a(x_6) = L_e^3(x_6) + L_e^4(x_6) + L_e^5(x_6) = 0 \qquad L_e^5(x_6) = L_e^3(x_6) + L_e^4(x_6) = +\infty$$

Irregular LDPC-Codes

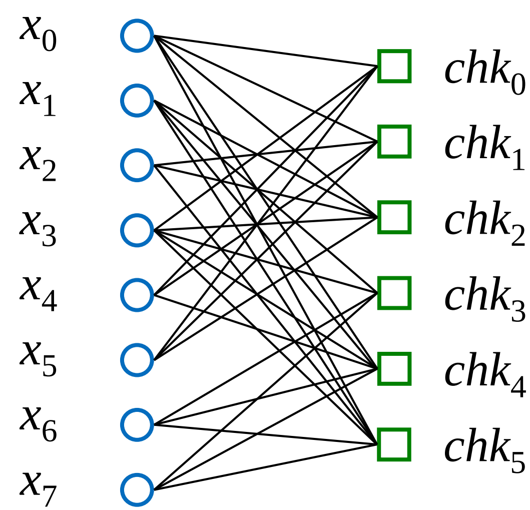
- Properties:
  - Generalization of regular LDPC codes
  - Lower error rates, i.e., better performance
  - Irregular number of ones per column and per row
  - Variable nodes of different degrees
  - Check nodes of different degrees
- Example:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$



Irregular LDPC-Codes

- Irregular number of ones per column and per row:
  - ℓ_i : proportion of left (variable) nodes of degree i
  - r_i : proportion of right (check) nodes of degree i
- In example:
  - $\ell_3 = 5/8$ $\ell_4 = 1/8$ $\ell_5 = 2/8$
  - $r_4 = 3/6$ $r_5 = 1/6$ $r_6 = 2/6$



- Proportions of edges:
  - λ_i : proportion of edges incident to left nodes of degree i
  - ρ_i : proportion of edges incident to right nodes of degree i
- In example:
  - $\lambda_3 = 15/29$ $\lambda_4 = 4/29$ $\lambda_5 = 10/29$
  - $\rho_4 = 12/29$ $\rho_5 = 5/29$ $\rho_6 = 12/29$

Irregular LDPC-Codes

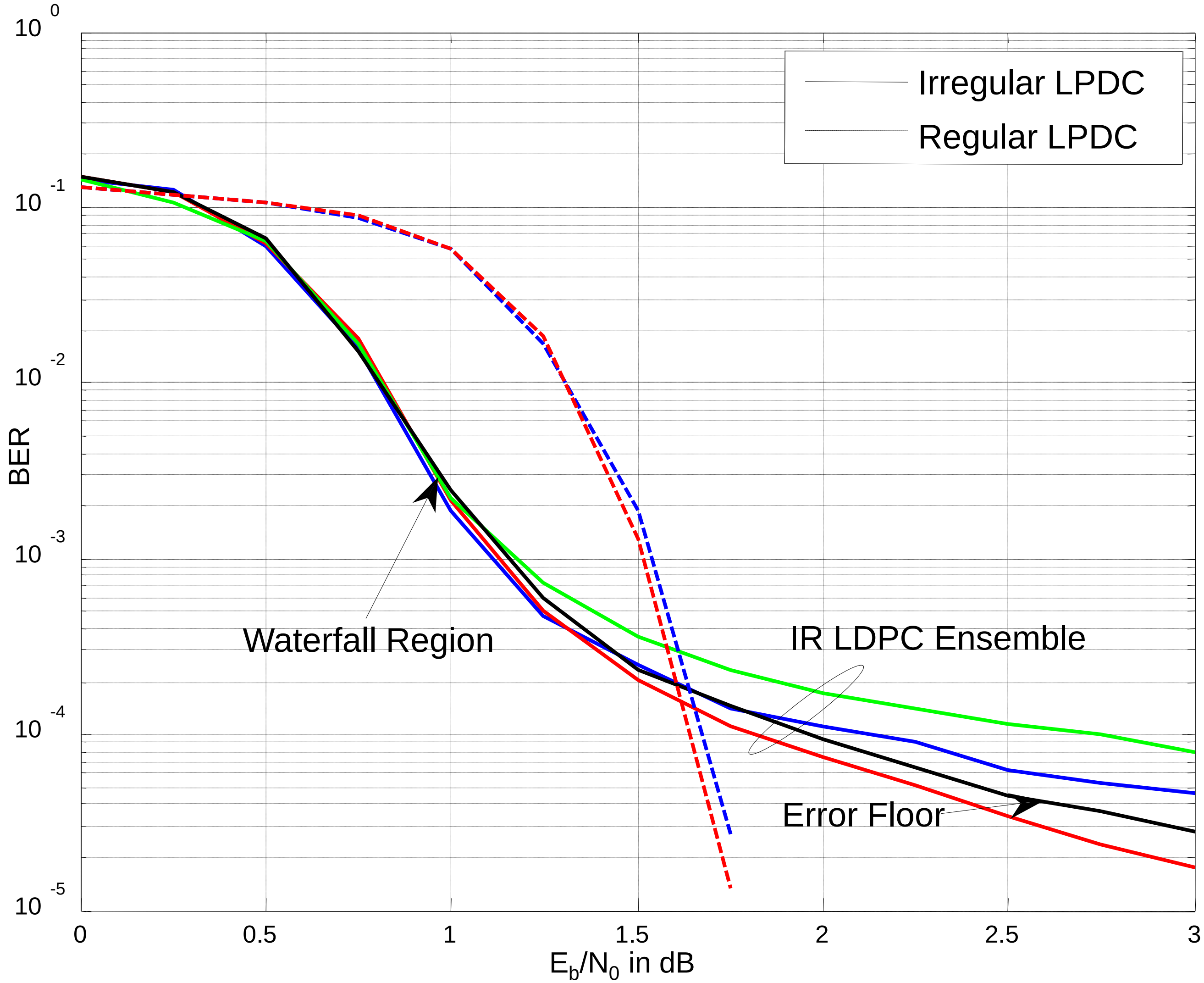
- LDPC codes are optimized via Density Evolution or EXIT analysis
  - Probability density functions describing the **distribution** of **check** and **variable** nodes in a parity check matrix
  - Specific codes can be found via random code generation following these distributions
  - → PDFs will only be nearly fulfilled due to the finite number of checks and variables
  - → Quality may vary in such an **ensemble** of codes due to random generation
- Example: R_c=1/2 LDPC Code with n=4096 and k=2048
  - Variable node distribution:

Degree	2	3	6	7	20
PDF	0.48394942887	0.29442753267	0.29442753267	0.074055964589	0.062432620582
Number	1986	1202	349	303	256

  - Check node distribution

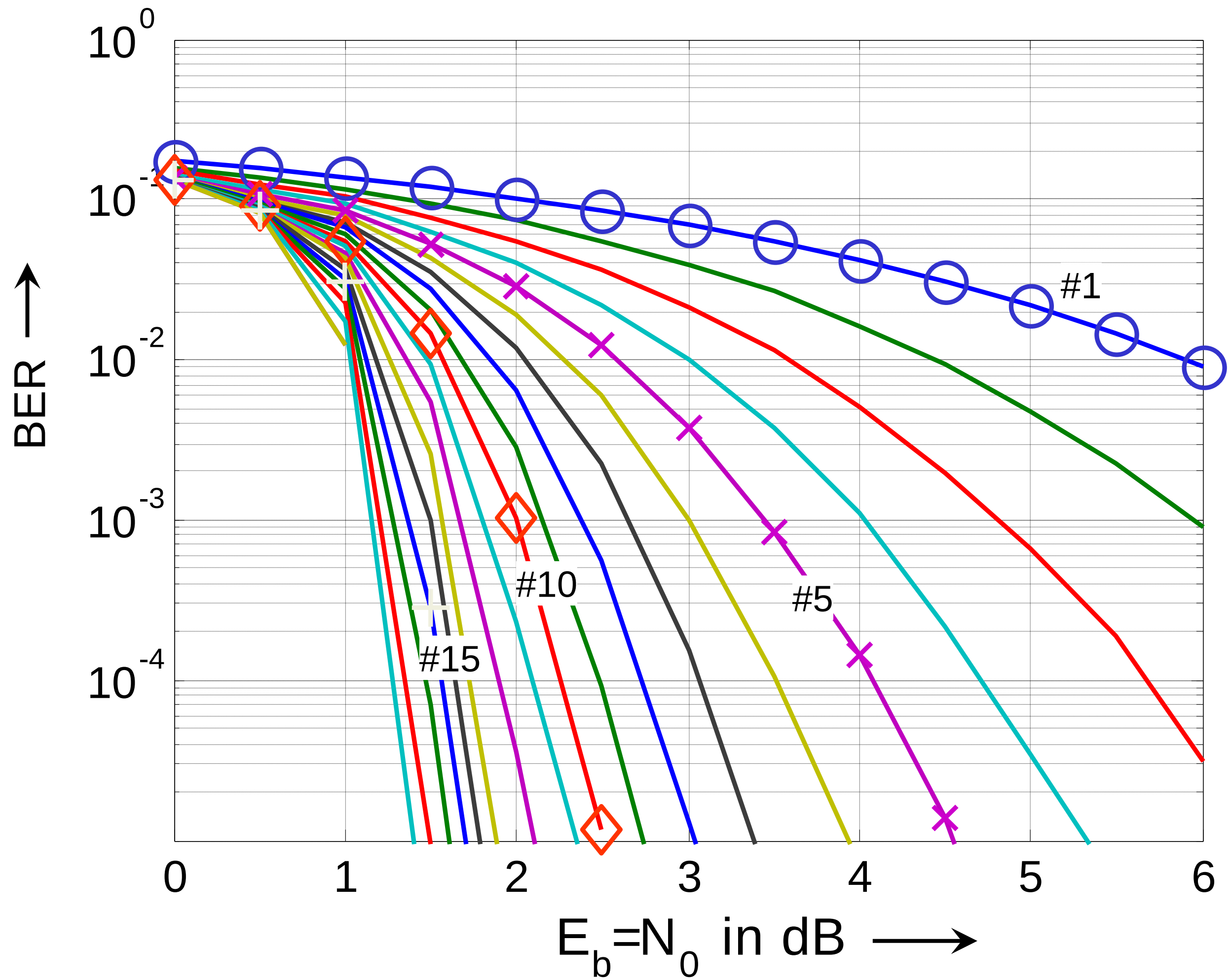
Degree	8	9
PDF	0.74193548387	0.25806451612
Number	1850	529

Simulation Results



- Irregular and regular LDPC code
  - IR as previous slide
  - Regular: n=4096, k=2048
    - 3 ones in a column
    - Random generation
- Performance
  - Irregular better in waterfall region
  - Error floor depends on n
  - → lower error floor possible
- Remarks
  - Regular codes are easier to attain

BER Performance of LDPC Code



- Number info bits
  $k = 9507$
- Code word length
  $N = 29507$
- Code rate
  $R_C = 0.322$