Convolution Code

CT-216 Introduction to Communication Systems

Project Group 9

May 6, 2025

Honor Code

We declare that:

- The work that we are presenting is our own work.
- We have not copied the work (the code, the results, etc.) that someone else has done.
- Concepts, understanding and insights we will be describing are our own.
- We make this pledge truthfully. We know that violation of this solemn pledge can carry grave consequences.

Team Members:

- 202301089 - YAKSH PATEL
- 202301090 - JHIL PATEL
- 202301091 - MEETKUMAR DIPAKKUMAR PATEL
- 202301092 - RAJDEEP PATEL
- 202301093 - NEEL SHAH
- 202301094 - YASHKUMAR PANCHAL
- 202301096 - SUTARIYA OM
- 202301097 - DEEP KAKADIYA
- 202301098 - KIRTAN CHHATBAR
- 202301099 - KRISH MALHOTRA

Contents

1 Overview of Convolutional Codes **5**
1.1 Key Characteristics . . . 5
1.2 Historical and Practical Significance . . . 5
1.3 Conceptual Operation . . . 6
1.4 Advantages . . . 6
1.5 Limitations . . . 6

2 Basic Parameters **7**

3 Encoder Structure **8**
3.1 Example of Encoding . . . 9

4 Encoder Representations **10**
4.1 Vector Form . . . 10
4.2 Polynomial Form . . . 10

5 State and Tree Diagrams **11**
5.1 Tree Diagram . . . 11
5.2 State Diagram . . . 12

6 Introduction to BPSK Modulation,AWGN Channel and BPSK Demodulation **14**

7 BPSK Modulator **14**
7.1 Concept . . . 14

8 AWGN Channel **15**
8.1 Concept . . . 15

9 BPSK Demodulator **16**
9.1 Concept . . . 16

10 Performance Analysis **17**
10.1 Theoretical Bit Error Rate (BER) . . . 17
10.2 Interpretation . . . 18

11 Decoder Structure **18**
11.1 Hard decoding Analysis 18
11.2 Soft decoding Analysis 21

12 Transfer Function of Convolutional Codes **26**

13 Conclusion **28**

14 References **31**

1 Overview of Convolutional Codes

Convolutional codes represent a distinct approach to error control coding, diverging from traditional block coding. The encoder processes the entire data stream into a continuous codeword by convolving the sequence of input bits with predefined generator sequences.

1.1 Key Characteristics

- Data is encoded sequentially without segmentation into fixed blocks.
- Encoding is performed using linear finite-state shift registers.
- The encoder operates as a memory-based system.
- Coding is governed by generator polynomials.

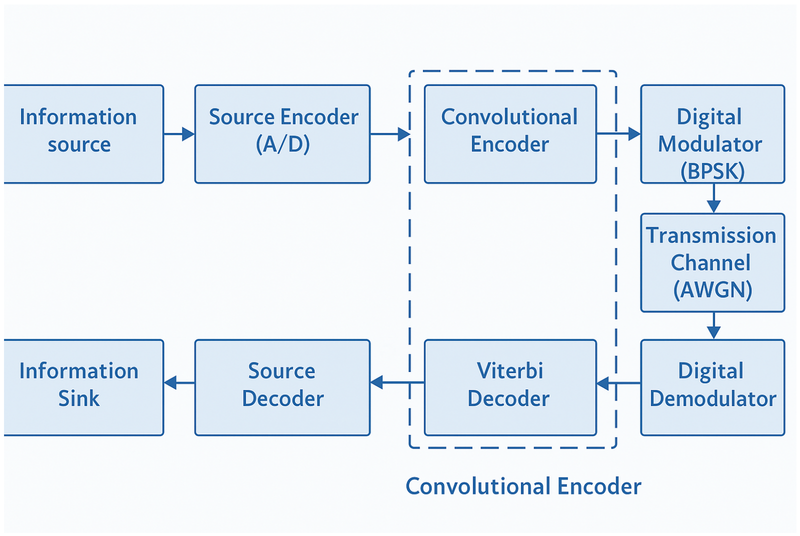


Figure 1: Flowchart of the Convolution Code

This design facilitates straightforward hardware implementation and is particularly effective for real-time communication systems.

1.2 Historical and Practical Significance

Convolutional coding was first introduced in the 1950s and gained significant attention in the 1960s with the development of efficient decoding algorithms, particularly the Viterbi algorithm. These codes are widely used in modern communication systems, including:

- Deep space communication (e.g., NASA missions)
- Mobile and cellular systems (2G, 3G, and LTE)
- Digital television and satellite broadcasting
- Wireless LANs and sensor networks

Their ability to provide strong error correction with relatively simple encoder hardware makes them a preferred choice in bandwidth- and power-constrained environments.

1.3 Conceptual Operation

The core idea behind convolutional encoding is to introduce structured redundancy into the data stream. Each output bit depends not only on the current input bit(s), but also on previous bits stored in the encoder's memory (shift registers). This temporal dependency allows the decoder to identify and correct burst errors more effectively than block codes that operate on isolated, fixed-length data blocks.

1.4 Advantages

- Continuous encoding process is ideal for streaming data.
- Strong error-correcting capability for relatively short memory lengths.
- Compatible with soft decision decoding (e.g., Viterbi algorithm), which significantly improves performance in noisy channels.
- Easily implemented in hardware using shift registers and XOR gates.

1.5 Limitations

- Increased decoding complexity for longer constraint lengths.
- Lower spectral efficiency compared to higher-rate block codes unless puncturing is applied.
- Requires tail bits (zero-padding) to terminate encoding, which adds overhead.

2 Basic Parameters

A convolutional code is defined by the tuple (n, k, K), where:

- k: Number of input bits.
- n: Number of output bits.
- K: Constraint length (depth of memory $= K - 1$).

The code rate is given by:

$$R_c = \frac{k}{n}$$

The performance of a convolutional code depends on the coding rate and the constraint length,

Longer constraint length K

- More powerful code
- More coding gain

Coding gain: the measure in the difference between the signal to noise ratio (SNR) levels between the uncoded system and coded system required to reach the same bit error rate (BER) level

- More complex decoder
- More decoding delay
- Smaller coding rate Rc=k/n
- More powerful code due to extra redundancy
- Less bandwidth efficiency

3 Encoder Structure

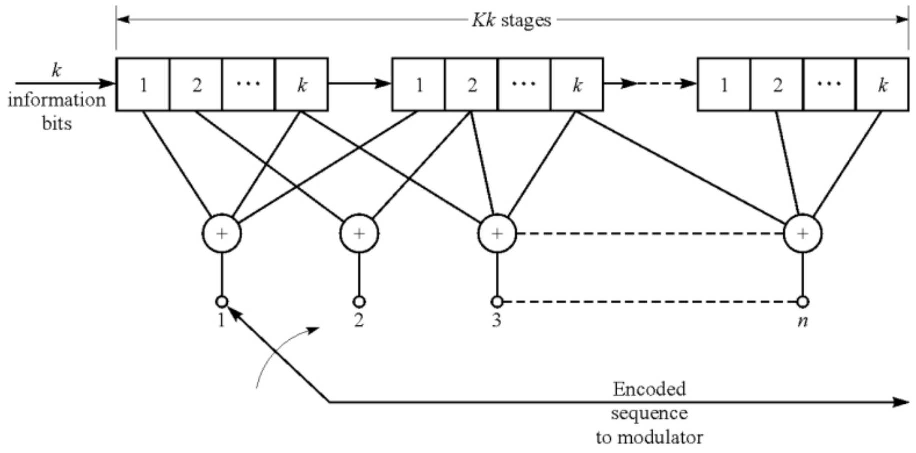


Figure 2: General Convolutional Encoder

An encoder with a rate 1/2 and constraint length $K = 3$ uses 3 shift registers. The current input and previous memory bits determine two output bits per input bit.

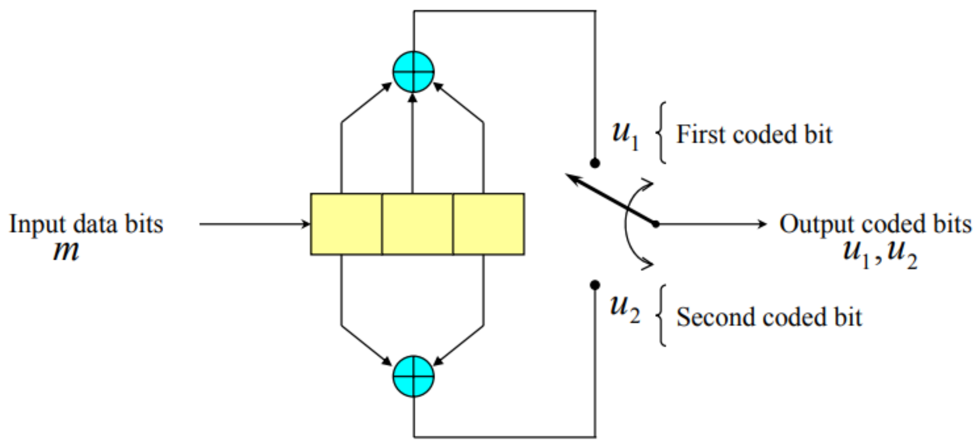
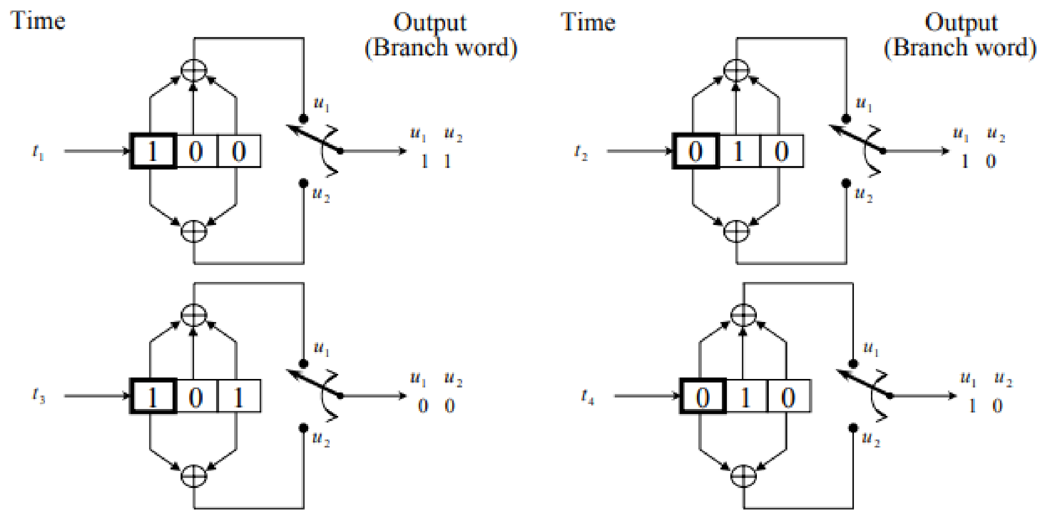


Figure 3: Example: Convolutional Encoder with Rate 1/2, $K = 3$

3.1 Example of Encoding

For message $m = (1, 0, 1)$ and generator polynomials $g_1 = 111$, $g_2 = 101$:

$$U = (11, 10, 00, 10, 11)$$



The effective code rate after zero-padding the input to clear the memory is:

$$R_{\text{eff}} = \frac{L}{n(L + K - 1)}$$

For $L = 3$, $n = 2$, $K = 3$: $R_{\text{eff}} = \frac{3}{2(5)} = 0.3$

4 Encoder Representations

4.1 Vector Form

Each modulo-2 adder is represented by a binary vector. For $g_1 = (1, 1, 1)$ and $g_2 = (1, 0, 1)$:

$$g_1 = [1\ 1\ 1], \quad g_2 = [1\ 0\ 1]$$

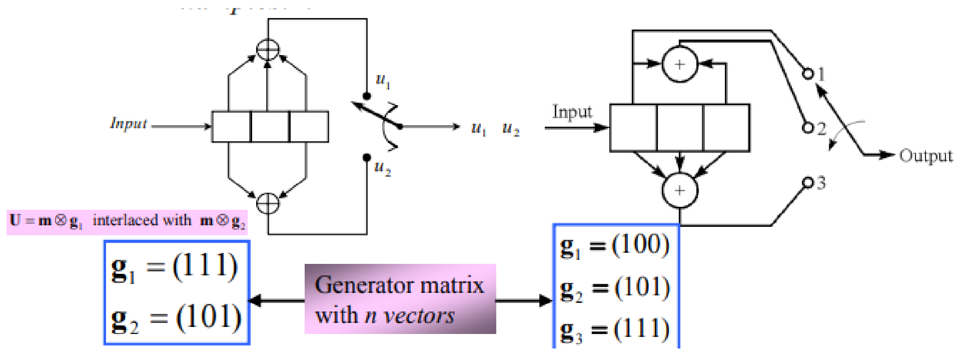


Figure 4: Example: Vector Form

4.2 Polynomial Form

Generator polynomials describe the shift register configuration:

$$g_1(X) = 1 + X + X^2, \quad g_2(X) = 1 + X^2$$

The output sequence is generated by multiplying the input polynomial $m(X)$ with each $g_i(X)$.

$$\mathbf{m}(X)\mathbf{g}_1(X) = (1+X^2)(1+X+X^2) = 1+X+X^3+X^4$$
$$\mathbf{m}(X)\mathbf{g}_2(X) = (1+X^2)(1+X^2) = 1+X^4$$
$$\mathbf{m}(X)\mathbf{g}_1(X) = 1+X+0.X^2+X^3+X^4$$
$$\mathbf{m}(X)\mathbf{g}_2(X) = 1+0.X+0.X^2+0.X^3+X^4$$
$$\mathbf{U}(X) = (1,1)+(1,0)X+(0,0)X^2+(1,0)X^3+(1,1)X^4$$
$$\mathbf{U} = 11 \quad 10 \quad 00 \quad 10 \quad 11$$

Figure 5: Example: m(1,0,1) Polynomial Form

5 State and Tree Diagrams

5.1 Tree Diagram

A tree diagram visualizes all paths from an initial state through sequential input bits. Each node corresponds to a state; edges represent input transitions.
K=3, k=1, n=3 convolutional encoder, The state of the first (K-1)*k stages of the shift register:
a=00; b=01;
c=10; d=11

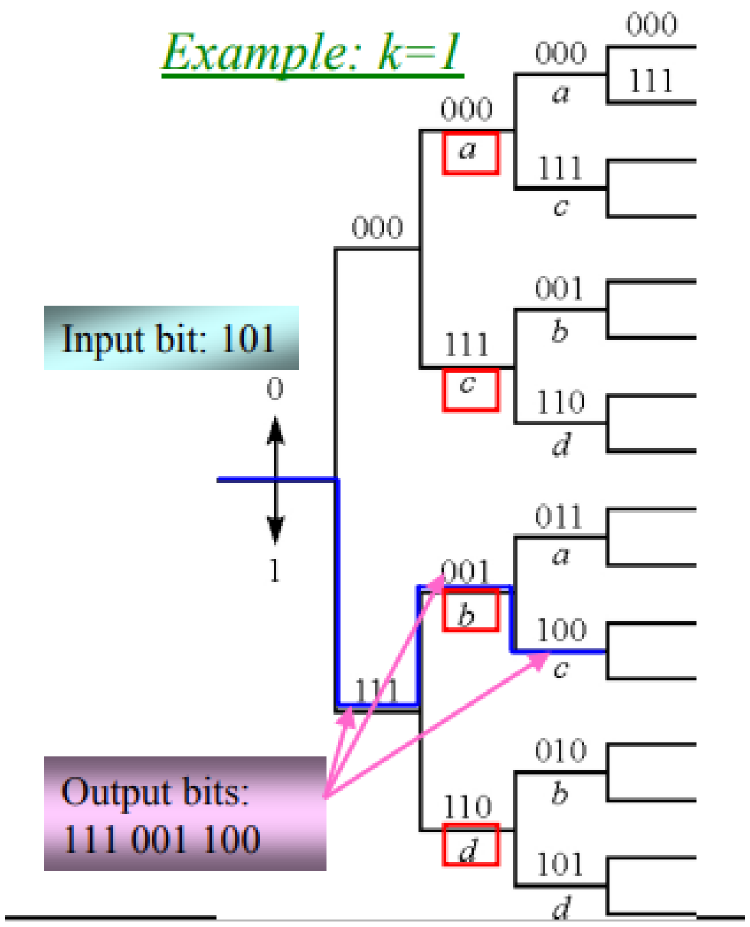


Figure 6: Example Tree Diagram for Input Sequence 101

5.2 State Diagram

A state diagram shows the finite-state machine behavior of the encoder. Each state is labeled by its memory content. Transitions are labeled with outputs and occur based on current input and memory.

A state diagram is simply a graph of the possible states of the encoder and the possible transitions from one state to another. It can be used to show the relationship between the encoder state, input, and output.

- The stage diagram has 2*(K-1)*k nodes, each node standing for one encoder state.
- Nodes are connected by branches

Every node has 2*k branches entering it and 2*k branches leaving it. The branches are labeled with c, where c is the output. When k=1

- The solid branch indicates that the input bit is 0.
- The dotted branch indicates that the input bit is 1.

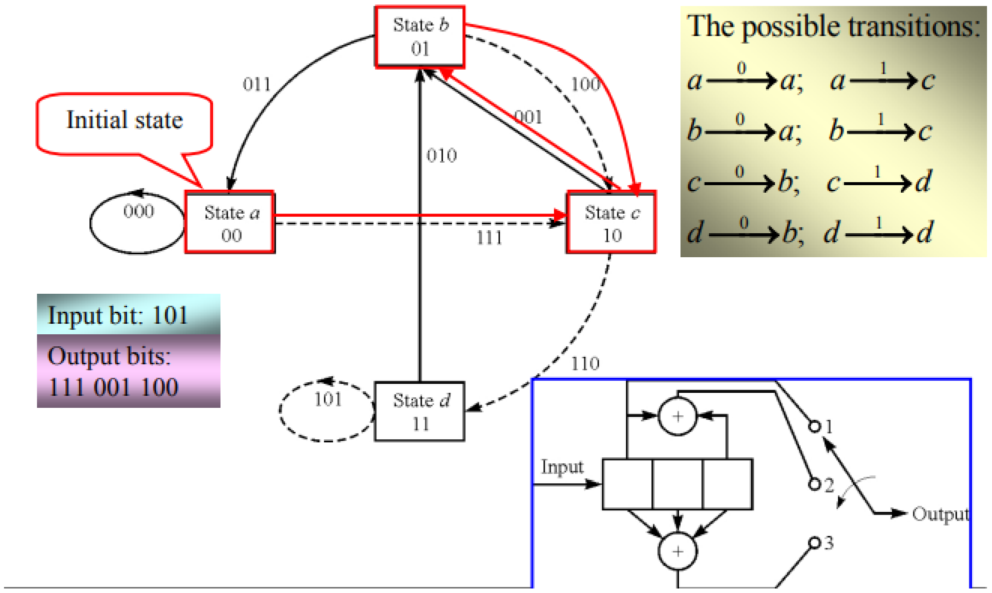

State b
01
State a
00
State c
10
State d
11
Initial state
000
011
100
001
010
111
110
101
The possible transitions:
$a \xrightarrow{0} a$; $a \xrightarrow{1} c$
$b \xrightarrow{0} a$; $b \xrightarrow{1} c$
$c \xrightarrow{0} b$; $c \xrightarrow{1} d$
$d \xrightarrow{0} b$; $d \xrightarrow{1} d$
Input bit: 101
Output bits:
111 001 100
Input
1
2
3
Output

Figure 7: State Diagram for a 4-State Encoder ($K = 3$)

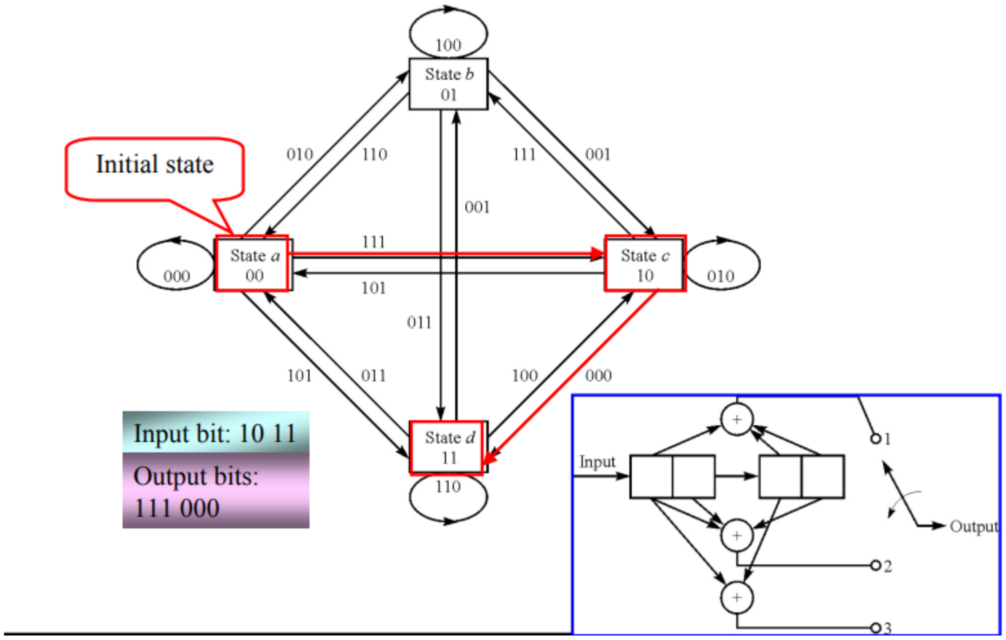


Figure 8: State Diagram for a 4-State Encoder ($K = 3$)

6 Introduction to BPSK Modulation,AWGN Channel and BPSK Demodulation

In modern digital communication systems, modulation and channel modeling are crucial for robust data transmission. Binary Phase Shift Keying (BPSK) is one of the simplest and most robust digital modulation schemes. It is commonly used in systems where bit error performance is critical. The Additive White Gaussian Noise (AWGN) channel models random noise that affects the transmitted signal.

7 BPSK Modulator

7.1 Concept

BPSK maps each binary bit to one of two phases:

$$s(t) = \begin{cases} +1, & \text{if } b = 0 \\ -1, & \text{if } b = 1 \end{cases}$$

This can be represented as:

$$s_i = 1 - 2b_i$$

Where 0 maps to +1 and 1 maps to -1.

Binary PSK (BPSK) is the simplest form of Phase Shift Keying. In BPSK, each bit is represented by a sinusoidal carrier wave with two different phases. The pair of signals $S_1(t)$ and $S_2(t)$ are used to represent binary symbols 1 and 0, respectively.

$$S_1(t) = \sqrt{\frac{2E_b}{T_b}} \cos(2\pi f_c t) \quad \text{(for bit 1)}$$

$$S_2(t) = \sqrt{\frac{2E_b}{T_b}} \cos(2\pi f_c t + \pi) = -\sqrt{\frac{2E_b}{T_b}} \cos(2\pi f_c t) \quad \text{(for bit 0)}$$

where:

- $0 \leq t \leq T_b$
- E_b is the energy per bit
- T_b is the bit duration
- f_c is the carrier frequency

8 AWGN Channel

8.1 Concept

Additive White Gaussian noise channel is the most common type of noise added over the channel. It is white because it has a constant power spectral density. It is Gaussian because its probability density function can be accurately modeled to behave like a Gaussian distribution
The AWGN channel adds white Gaussian noise to the transmitted signal. The received signal is:

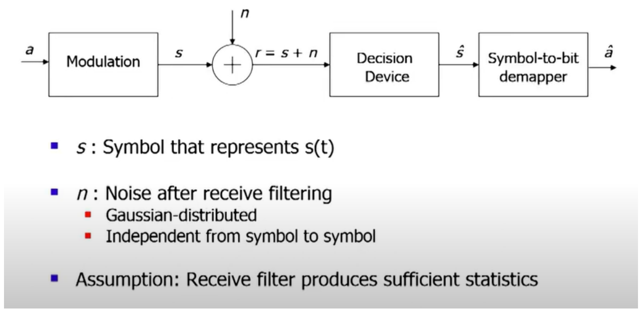
$$r = s + n$$

where $n_i \sim \mathcal{N}(0, \sigma^2)$ is the Gaussian noise with zero mean and variance σ^2. The noise variance is related to the SNR as:

$$\sigma^2 = \frac{1}{2R \cdot \text{SNR}_{\text{linear}}}$$

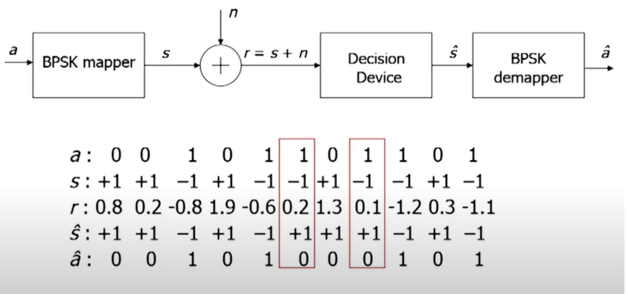
where R is the code rate (e.g., 1 for uncoded BPSK), and

$$\text{SNRlinear} = 10^{\text{SNRdB}/10}$$



- s : Symbol that represents s(t)
- n : Noise after receive filtering
  - Gaussian-distributed
  - Independent from symbol to symbol
- Assumption: Receive filter produces sufficient statistics

Figure 9: AWGN Channel



a:	0	0	1	0	1	1	0	1	1	0	1
s:	+1	+1	−1	+1	−1	−1	+1	−1	−1	+1	−1
r:	0.8	0.2	-0.8	1.9	-0.6	0.2	1.3	0.1	-1.2	0.3	-1.1
$\hat{s}$:	+1	+1	−1	+1	−1	+1	+1	+1	−1	+1	−1
$\hat{a}$:	0	0	1	0	1	0	0	0	1	0	1

Figure 10: BPSK over AWGN

9 BPSK Demodulator

9.1 Concept

The receiver performs coherent demodulation assuming perfect synchronization. The decision rule is:

$$\hat{b}_i = \begin{cases} 0, & \text{if } r_i \geq 0 \\ 1, & \text{if } r_i < 0 \end{cases}$$

At the receiver, the received signal undergoes demodulation to recover the original binary data. This process typically involves:
• Coherent Detection: Multiplying the received signal with a local oscillator at the carrier frequency to extract the phase information.
• Decision Making: Comparing the phase of the received signal

with a threshold value to determine the transmitted symbol. If the phase is closer to one phase state, it's interpreted as '0';if it's closer to the other phase state, it's interpreted as '1'.
For soft decision decoding
• If the convolutional code produces p parity bits, and the p corresponding analog samples are v = v1, v2,...,vp.
• These values are analog in nature as we do not perform demodulation of the received signal we directly pass it to the decoder.

10 Performance Analysis

10.1 Theoretical Bit Error Rate (BER)

Step 1: Signal and Noise

Let the transmitted symbol be $s \in \{+1, -1\}$. The received signal is:

$$r = s + n$$

where $n \sim \mathcal{N}(0, \sigma^2)$ is AWGN.

Step 2: Decision Rule

The receiver decides:

$$\hat{s} = \begin{cases} +1 & \text{if } r \geq 0 \\ -1 & \text{if } r < 0 \end{cases}$$

Step 3: Probability of Error

Assume $s = +1$ was transmitted. An error occurs if $r < 0$:

$$P_e = P(r < 0 | s = +1) = P(n < -1)$$

Step 4: Using Q-function

The Q-function is defined as:

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-t^2/2} dt$$

So,

$$P_e = Q\left(\frac{1}{\sigma}\right)$$

Step 5: Expressing in terms of SNR

Since $\sigma^2 = \frac{N_0}{2}$, then:

$$P_b = Q\left(\sqrt{\frac{2E_b}{N_0}}\right)$$

10.2 Interpretation

The BER decreases exponentially as the SNR increases. BPSK is particularly effective at maintaining low BER at low SNR levels, making it ideal for noisy communication environments.

11 Decoder Structure

The decoding of convolutional codes can be done in two primary ways. These methods differ in how they interpret received data and affect decoding performance.

11.1 Hard decoding Analysis

- We will analyze the performance of maximum likelihood decoding for a convolutional code over a binary symmetric channel (BSC).

- Without loss of generality, we assume that the all-zero codeword $\mathbf{0}$ is transmitted.
- A first event error occurs at an arbitrary time t if the all-zero path is eliminated for the first time in favor of an incorrect path.
- Assuming that the incorrect path has Hamming weight d, a first event error occurs with probability:

$$P_2(D) = \begin{cases} \sum\limits_{k=(d+1)/2}^{d} \binom{d}{k} p^k (1-p)^{d-k} & \text{If } d \text{ is odd} \quad \text{....(a)} \\ \sum\limits_{k=(d+1)/2}^{d} \binom{d}{k} p^k (1-p)^{d-k} + \frac{1}{2}\binom{d}{d/2} p^{d/2}(1-p)^{d/2} & \text{If } d \text{ is even} \quad \text{....(b)} \end{cases}$$

- Instead of using the expressions for $P_2(d)$ given in (a) and (b), we can use the upper bound,

$$P_2(D) < [4p(1-p)]^{d/2} \quad \text{....(c)}$$

- Use of this bound in (c) yields a looser upper bound on the first-event error probability, in the form,

$$P_e < \sum_{d=d_{\text{free}}}^{d} a_d [4p(1-p)]^{d/2}$$
$$< T(D)\Big|_{D=\sqrt{4p(1-p)}} \quad \text{....(d)}$$

For odd d,

$$P_2(d) = \sum_{e=\frac{d+1}{2}}^{d} \binom{d}{e} p^e (1-p)^{d-e}$$
$$\leq \sum_{e=\frac{d+1}{2}}^{d} \binom{d}{e} p^{d/2} (1-p)^{d/2}$$

$$= p^{d/2}(1-p)^{d/2} \sum_{e=0}^{d} \binom{d}{e}$$
$$= 2^d p^{d/2}(1-p)^{d/2}$$

For even d,

$$\begin{aligned}
P_2(d) &= \sum_{e=\frac{d}{2}+1}^{d} \binom{d}{e} p^e (1-p)^{d-e} + \frac{1}{2}\binom{d}{\frac{d}{2}} p^{d/2}(1-p)^{d/2} \\
&< \sum_{e=\frac{d}{2}}^{d} \binom{d}{e} p^e (1-p)^{d-e} \\
&< \sum_{e=\frac{d}{2}}^{d} \binom{d}{e} p^{d/2}(1-p)^{d/2} \\
&= p^{d/2}(1-p)^{d/2} \sum_{e=0}^{d} \binom{d}{e} \\
&= 2^d p^{d/2}(1-p)^{d/2}
\end{aligned}$$

- The bit error probability can be bounded by

$$P_b < \sum_{d=d_{\text{free}}}^{\infty} \beta_d P_2(d) \qquad \text{....(e)}$$

- where β_d is the total number of nonzero information bits on all weight-d paths, divided by the number of information bits k per unit time.

- The β_d are the coefficients in the expansion of the derivative of $T(D, N)$, evaluated at $N = 1$. For $P_2(d)$, we may use either the expressions given in (a) and (b) or the upper bound in (c). If the latter is used, the upper bound on P_b can be expressed as:

$$P_b < \left.\frac{dT(D,N)}{dN}\right|_{N=1,D=\sqrt{4p(1-p)}} \qquad ....(\mathrm{f})$$

- When $k > 1$, the results given in (e) and (f) for P_b should be divided by k.

11.2 Soft decoding Analysis

- Convolution Encoder is a finite state machine, and the optimum decoder is a maximum likelihood sequence estimator (MLSE), which helps us to find the most likely transmitted sequence for the received sequence.
- If the Soft-Decision Decoding is employed and the code sequence is transmitted by BPSK modulation then the input to the decoder is:

  $$r_{jm} = \sqrt{E_c}(2c_{jm} - 1) + \eta_{jm} \qquad .....(1)$$

  Here:

  - c_{jm} : coded symbol related to m^{th} symbol of j^{th} branch and its components are c_{jm1}, c_{jm2} etc.
  - r_{jm} : output from the demodulator i.e. the Viterbi decoder inputs corresponding to m^{th} symbol of j^{th} branch in trellis.
  - η_{jm} : represents the additive white Gaussian noise that affects the reception of m^{th} symbol of j^{th} branch in trellis.

  Branch Metric:
  A branch metric for the j^{th} branch of the i^{th} path through the trellis is defined as the logarithm of the joint probability of the sequence conditioned on transmitted sequence:

  $$\mu_j^{(i)} = \log P\left(\{r_{jm}\} \mid \{c_{jm}\}\right) \qquad .....(2)$$

Now,

$$P(e_m \mid \{C_j\}) = P(T_{m1}, T_{m2}, \ldots, T_{m3} \mid C_{jm1}, C_{jm2}, C_{jm3}, \ldots)$$

Since T_j's are independent of each T_j for any $i \neq m$,

$$P(e_{jm} \mid \{C_{jm'}\}) = \prod_n P(t_{ml} \mid C_{jm'})$$

Considering that the reception of T_{ij} is only being affected by C_{jm},

$$P(t_{ml} \mid G_{jm'}) = P(t_{ml} \mid C_{jm'})$$

Now, demodulator output is described statistically by the PDF:

$$P(T_{im} \mid C_{im}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(T_{jm} - \sqrt{E_c}(2c_{jm} - 1))^2}{2\sigma^2}\right) \qquad .....(3)$$

Where $\sigma^2 = N_0/2$, the variance of Additive White Gaussian Noise.

Using equations (a), (b) into equation (2), and by neglecting the common terms to all branch metrics, the branch metric for the j^{th} branch of the i^{th} path may be expressed as:

$$H_j = \sum T_{jm}(2c_{jm} - 1)$$

Path Metric

A path metric for the i^{th} path containing B branches through the trellis is defined as:

$$PM^{(i)} = \sum_j H_j$$

The guidelines for choosing between two paths within the trellis is to opt for the one with higher metric. This enhances the likelihood of making accurate decisions or conversely, diminishes the probability of errors in the sequence of information bits.

It is also called the convolution metric of the path:

$$CM^{(i)} = \sum_{j} T_{jm}(2c_{jm'} - 1) \qquad .....(4)$$

This metric is essentially the summation of all branch metrics of the i^{th} path.

$$CM^{(0)} = \sqrt{E_c}Bn - \sum_{j=1}^{B}\sum_{m=1}^{n} \eta_{jm} \qquad .....(5)$$

- Now, If the incorrect path that merges with all-zero path differs from ‘d’ bits will be having its correlation metric $CM^{(1)}$, then the probability of error in pairwise comparison of $CM^{(0)}$ and $CM^{(1)}$ is:

  $$P_2(d) = P\left(CM^{(1)} \geq CM^{(0)}\right) = P\left(CM^{(1)} - CM^{(0)} \geq 0\right)$$

- Using eq(4)

  $$P_2(d) = P\left(2\sum_{j=1}^{B}\sum_{m=1}^{n} r_{jm}\left(c_{jm}^{(1)} - c_{jm}^{(0)}\right) \geq 0\right) \qquad .....(6)$$

- Since the coded bits in the two paths we identical except in the ‘d’ positions, this equation can be written as

  $$P_2(d) = P\left(\sum_{i=1}^{d} r'_l \geq 0\right) \qquad .....(7)$$

  Here, r'_l represents input to the decoder for d^{th} bits.
  we know that

  $$r_{jm} = \sqrt{E_c}(2c_{jm} - 1) + \eta_{jm}$$

  Where, $\eta_{jm} \sim \mathcal{N}\left(0, \frac{N_0}{2}\right) \Rightarrow r_{jm}/r'_l \sim \mathcal{N}\left(\sim \sqrt{E_c}, \frac{N_0}{2}\right)$

- Now, from Central Limit Theorem:

$$\sum_{i=1}^{d} r_l' \sim \mathcal{N}\left(-d\sqrt{E_c}, \frac{dN_0}{2}\right)$$

- Therefore,

$$P_2(d) = P\left[\frac{\sum_{i=1}^{d} r_l' + d\sqrt{E_c}}{\sqrt{\frac{dN_0}{2}}} \geq \frac{d\sqrt{E_c}}{\sqrt{\frac{dN_0}{2}}}\right] = Q\left(\sqrt{\frac{2dE_c}{N_0}}\right)$$

$$= Q\left(\sqrt{2\gamma_b R_c d}\right) \qquad .....(8)$$

- Where, $\gamma_b = \frac{E_b}{N_0}$ is received SNR per bit, and R_c is code rate.
- Now, there are many paths with different distances that merge with the all-zero path at a given node. Transfer function provides $T(D)$, a complex description of all such paths.
- Thus, we can sum the error probability for all such possible paths which will contribute to the first event error probability in the form:

$$P_e \leq \sum_{d=d_{\text{free}}}^{\infty} a_d P_2(d) \leq \sum_{d=d_{\text{free}}}^{\infty} a_d Q\left(\sqrt{2\gamma_b R_c d}\right) \qquad .....(9)$$

- Where,
  - d_{free} – is the smallest Hamming distance between any two particular codewords.
  - a_d – Number of paths of distance d (which particularly is the number of 1's in that path).

- Now, applying Chernoff's inequality we get $P(z \geq x) \leq \exp(-ax)E[e^{tX}]$, which further yields to the result:

$$Q(x) \leq \exp\left(-\frac{x^2}{2}\right)$$

$$Q\left(\sqrt{2\gamma_b R_c d}\right) \leq \exp(-\gamma_b R_c d) = D^d \quad \text{where } D = e^{-\gamma_b R_c} \qquad .....(10)$$

- From above equations :

$$P_e < \sum_{d=d_{\text{free}}}^{\infty} a_d D^d$$

- Now, bit error probability can be a more useful measure of performance. This probability can be upper bounded by the procedure used in bounding the first event error probability if we multiply $P_2(d)$ by the number of incorrectly decoded information bits.

- The average bit error probability is upper bounded by multiplying each $P_2(d)$ by the corresponding number of incorrectly decoded bits for each possible incorrect path.

- Now, $T(D, N) = \sum_{d=d_{\text{free}}}^{\infty} a_d D^d N^{f(d)}$ .....(11). Here, exponent of N represents the number of 1's in the path with d distance.

- Therefore,

$$\frac{T(D, N)}{dN} = \sum_{d=d_{\text{free}}}^{\infty} a_d f(d) D^d$$

$$\frac{T(D, N)}{dN} = \sum_{d=d_{\text{free}}}^{\infty} \beta_d D^d \qquad .....(12)$$

- Thus, the bit error probability for $k = 1$ is:

$$P_b < \sum_{d=d_{\text{free}}}^{\infty} \beta_d P_2(d)$$

$$P_b < \sum_{d=d_{\text{free}}}^{\infty} \beta_d Q\left(\sqrt{2\gamma_b R_c d}\right) \qquad .....(13)$$

12 Transfer Function of Convolutional Codes

To analyze distance properties of the code, the state diagram is transformed by:

- Splitting the initial and final all-zero state.
- Labeling each branch by its Hamming weight.

The transfer function $T(D)$ is defined as:

$$T(D) = \sum a_d D^d$$

where a_d is the number of paths of weight d. The smallest d for which $a_d \neq 0$ is the **free distance** d_{free}.

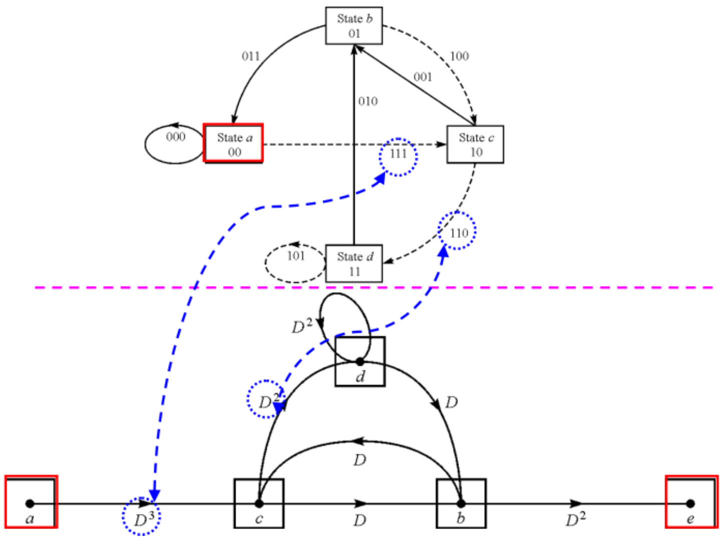
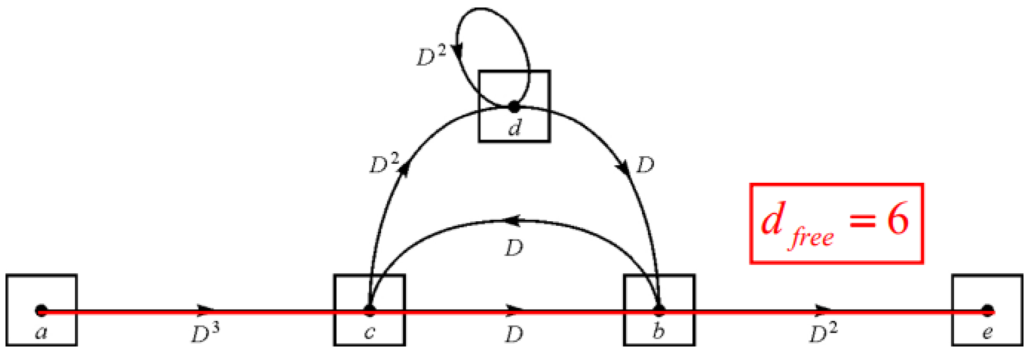


Figure 11: Modified State Diagram to Derive Transfer Function



$$
\begin{aligned}
X_c &= D^3 X_a + D X_b \\
X_b &= D X_c + D X_d \\
X_d &= D^2 X_c + D^2 X_d \\
X_e &= D^2 X_b
\end{aligned}
$$

$$
\begin{aligned}
T(X) = X_e / X_a &= D^6 / (1 - 2D^2) \\
&= D^6 + 2D^8 + 4D^{10} + 8D^{12} + \cdots \\
&= \sum_{d=6}^{\infty} a_d D^d
\end{aligned}
$$

$$
a_d = \begin{cases} 2^{(d-6)/2} & (\text{even } d) \\ 0 & (\text{odd } d) \end{cases}
$$

Figure 12: Transfer Function Example

13 Conclusion

- Convolutional codes improve reliability by introducing redundancy using parity bits instead of directly transmitting message bits.
- Binary Phase Shift Keying (BPSK) is a robust and simple modulation scheme that performs well in noisy environments.
- Transmission over an Additive White Gaussian Noise (AWGN) channel introduces random noise that degrades signal integrity.
- Bit Error Rate (BER) analysis demonstrates that performance improves with increasing signal-to-noise ratio (SNR).
- BPSK over AWGN provides a foundational model for analyzing and understanding digital communication systems.
- Hard Decision Decoding (HDD) offers a low-complexity approach by making binary decisions at the receiver, but may discard valuable amplitude information.
- Soft Decision Decoding (SDD), while more computationally intensive, significantly improves BER performance by leveraging the full amplitude information of the received signals.
- Simulation results show that SDD consistently achieves better BER performance than HDD, especially at lower E_b/N_0 values.
- Among the evaluated schemes, the 1/3 rate convolutional code with constraint length $K_c = 6$ and SDD provides the best BER performance.

- Decoder complexity increases with constraint length due to the exponential growth in trellis states in the Viterbi algorithm.
- SDD also adds complexity by requiring finer quantization and more computations per bit compared to HDD.
- These enhancements make convolutional coding with SDD ideal for systems where reliability is more critical than decoding complexity, such as in deep-space or mobile communications.
- In resource-constrained systems (e.g., IoT devices), HDD or lower constraint lengths might be preferred to maintain acceptable power and latency.
- The coding gain from convolutional coding with SDD becomes especially noticeable at moderate to high SNR ranges, offering significant improvements in BER.
- The trade-off between complexity and performance should guide system designers in selecting appropriate codes for specific applications.
- Future work could explore the use of turbo codes or LDPC codes, which offer even better performance with iterative decoding, albeit at further complexity cost.

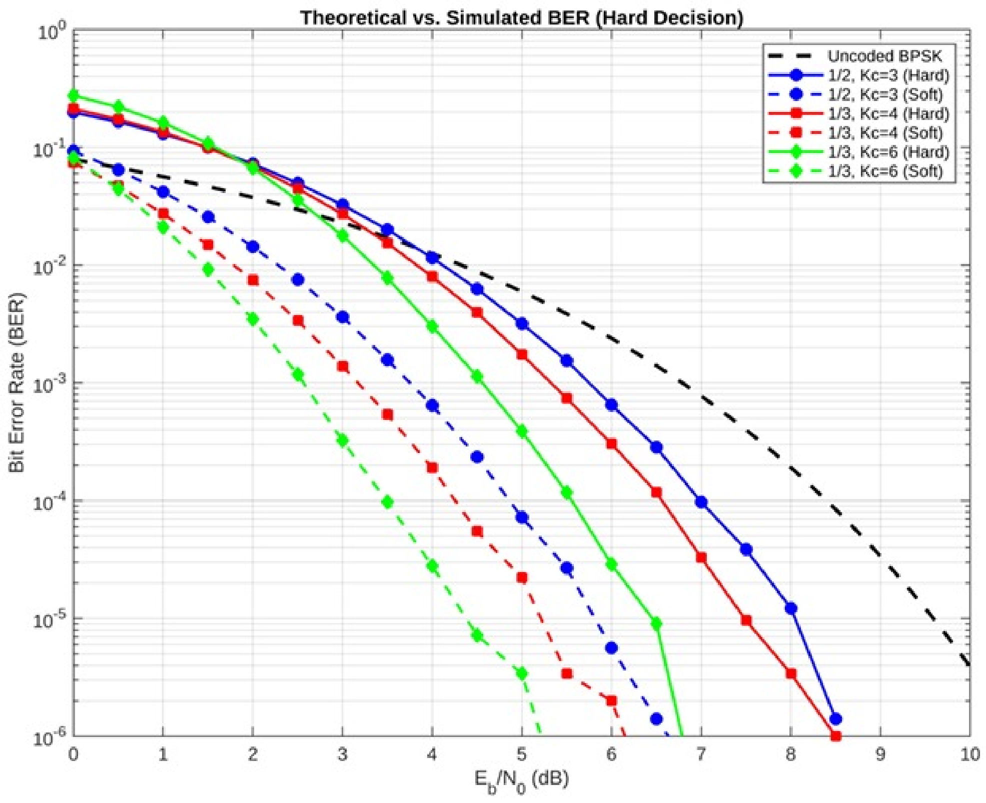


Figure 13: Theoretical vs. Simulated BER

14 References

1.Subhramanyam K N Video Lectures on Youtube:
link for playlist
2.Convolutional Codes and 'Their Performance in Communication Systems by ANDREW J. VITERBI, SENIOR MEMBER, IEEE
3.Article of The University of Hong Kong (HKU) on Convolution code.
4.Additive White Gaussian Noise(AWGN) Channel and BPSK
5.BER Analysis of BPSK for Convolution Codes Over AWGN Channel