

Lecture 8: Convolutional codes.

Course instructor: Alexey Frolov

`al.frolov@skoltech.ru`

Teaching Assistant: Stanislav Kruglik

`stanislav.kruglik@skolkovotech.ru`

February 15, 2018

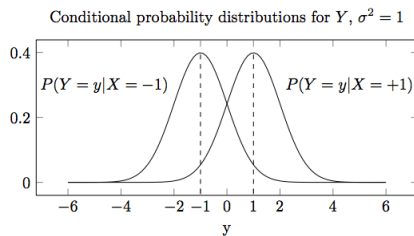
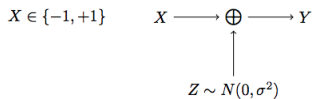
- 1 Soft decoding
- 2 First look at convolutional codes
- 3 Termination methods
- 4 Algebraic description
- 5 Distance properties
- 6 Decoding

The class of convolutional codes was invented by Elias in 1955 and has been widely in use for wireless, space, and broadcast communications since about 1970. Their popularity stems from the relative ease with which the *maximum-likelihood sequence decoder* may be implemented and from their effectiveness when concatenated with a ReedSolomon code.

- 1 Soft decoding
- 2 First look at convolutional codes
- 3 Termination methods
- 4 Algebraic description
- 5 Distance properties
- 6 Decoding

In case of *hard decision decoding* demodulator makes a hard decision and provides the decoder with the symbols of the field. In case of *soft decision decoding* demodulator provides the decoder with probability distributions.

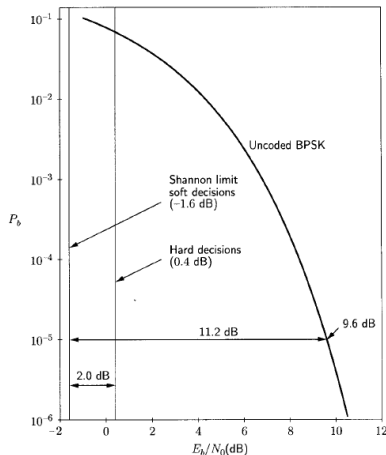
Example: Binary input AWGN channel



Assume again, that 0 is sent with the level +1 and 0 is sent with the level -1. Let us calculate the log likelihood ratio given y

$$\begin{aligned} LLR &= \log \frac{\Pr(X = +1|y)}{\Pr(X = -1|y)} \\ &= \log \frac{\Pr(y|X = +1) \Pr(X = +1) \Pr(y)}{\Pr(y|X = -1) \Pr(X = -1) \Pr(y)} \\ &= \log \frac{f(y|+1)}{f(y|-1)} \\ &= \frac{(y+1)^2 - (y-1)^2}{2\sigma^2} \\ &= \frac{2}{\sigma^2} y \end{aligned}$$

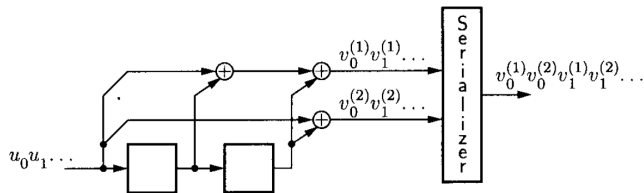
Soft vs hard decoding



Consider again BI-AWGN channel. We can use LLR values or make a hard decision (+1 if $LLR > 0$, -1 otherwise). The difference is approx. 2 dB!

- 1 Soft decoding
- 2 First look at convolutional codes**
- 3 Termination methods
- 4 Algebraic description
- 5 Distance properties
- 6 Decoding

First look at convolutional codes



An encoder for binary rate $R = 1/2$ convolutional code.

First look at convolutional codes

Infinite sequence of information digits

$$\mathbf{u} = u_0 u_1 \dots$$

The register length is called a *memory* of convolutional code.

Two output sequences (also infinite)

$$\mathbf{v}^{(1)} = v_0^{(1)} v_1^{(1)} \dots$$

and

$$\mathbf{v}^{(2)} = v_0^{(2)} v_1^{(2)} \dots$$

General rate $R = b/c$ convolutional encoder

Information sequence:

$$\mathbf{u} = \mathbf{u}_0 \mathbf{u}_1 \dots = u_0^{(1)} u_0^{(2)} \dots u_0^{(b)} u_1^{(1)} u_1^{(2)} \dots u_1^{(b)} \dots$$

Encoded sequence:

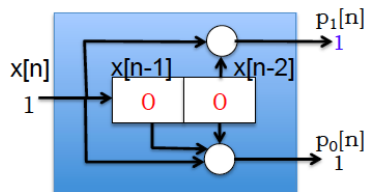
$$\mathbf{v} = \mathbf{v}_0 \mathbf{v}_1 \dots = v_0^{(1)} v_0^{(2)} \dots v_0^{(c)} v_1^{(1)} v_1^{(2)} \dots v_1^{(c)} \dots,$$

where

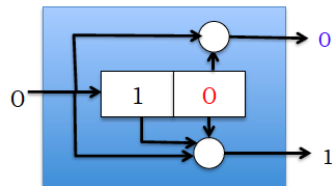
$$\mathbf{v}_t = f(\mathbf{u}_t, \mathbf{u}_{t-1}, \dots, \mathbf{u}_{t-m}).$$

Recall, that m is the encoder memory.

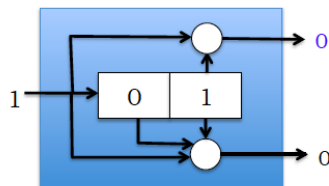
Example: transmit message 1011



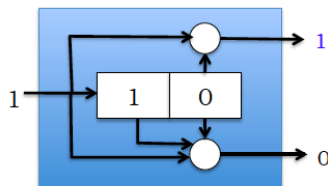
Processing $x[0]$



Processing $x[1]$



Processing $x[2]$



Processing $x[3]$

General rate $R = b/c$ convolutional encoder

The function f is a linear function (our requirement), thus

$$\mathbf{v}_t = \mathbf{u}_t \mathbf{G}_0 + \mathbf{u}_{t-1} \mathbf{G}_1 + \dots + \mathbf{u}_{t-m} \mathbf{G}_m,$$

where G_i , $0 \leq i \leq m$ is a binary $b \times c$ matrix.

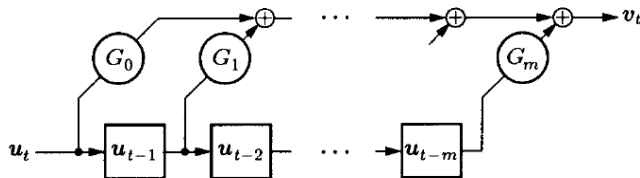
The encoding can be presented as follows

$$\mathbf{v} = \mathbf{uG}.$$

$$\mathbf{G} = \begin{pmatrix} \mathbf{G}_0 & \mathbf{G}_1 & \dots & \mathbf{G}_m & & \\ & \mathbf{G}_0 & \mathbf{G}_1 & \dots & \mathbf{G}_m & \\ & & \ddots & \ddots & & \ddots \end{pmatrix}$$

Generator matrix

\mathbf{G} is called a generator matrix, G_i , $0 \leq i \leq m$, – generator submatrices.



Example

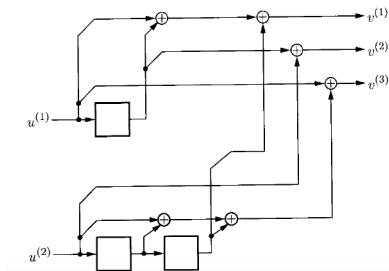
$$G_0 = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

$$G_1 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

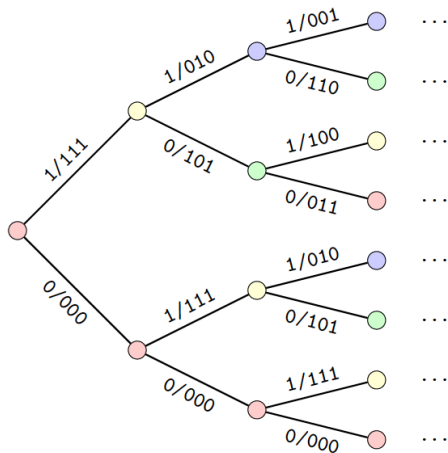
$$G_2 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

The generator matrix is

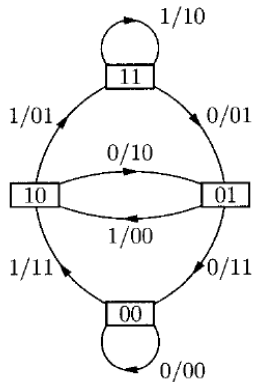
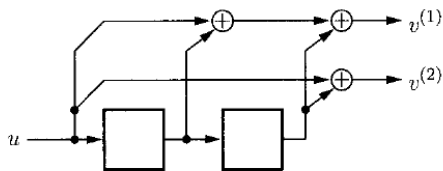
$$G = \begin{pmatrix} 101 & 110 & 000 & & \\ 011 & 001 & 101 & & \\ & 101 & 110 & 000 & \\ & 011 & 001 & 101 & \\ & & \ddots & \ddots & \ddots \end{pmatrix}$$



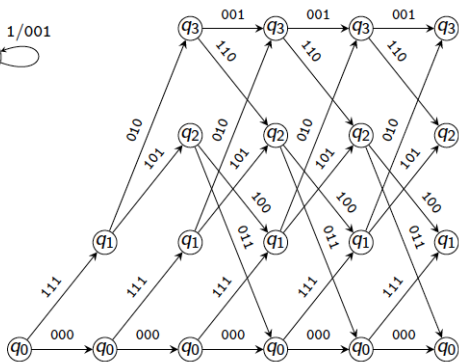
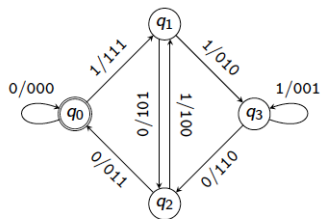
Code tree



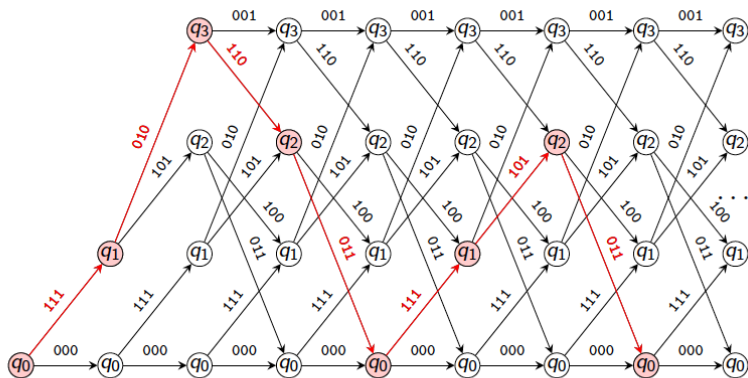
Finite state machine representation



Trellis



Example



1100100... \mapsto 111 010 110 011 111 101 011...

- 1 Soft decoding
- 2 First look at convolutional codes
- 3 Termination methods**
- 4 Algebraic description
- 5 Distance properties
- 6 Decoding

Termination methods

- ① Direct termination
- ② Zero termination
- ③ Tail biting

- 1 Soft decoding
- 2 First look at convolutional codes
- 3 Termination methods
- 4 Algebraic description**
- 5 Distance properties
- 6 Decoding

Convolutional codes are often thought of as nonblock codes over a finite field, but it can be an advantage to treat them as block codes over an infinite field.

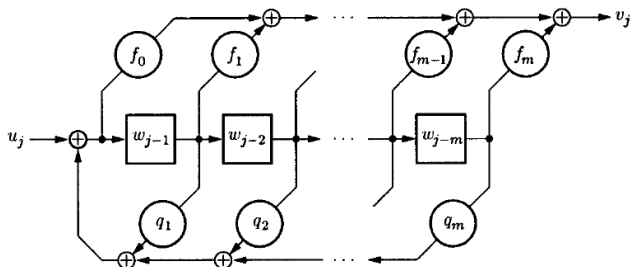
D - delay operator.

Representation of input and output sequences (D -transform)

$$\mathbf{u}(D) = \dots \mathbf{u}_{-1}D^{-1} + \mathbf{u}_0 + \mathbf{u}_1D + \mathbf{u}_2D^2 + \dots$$

$$\mathbf{v}(D) = \dots \mathbf{v}_{-1}D^{-1} + \mathbf{v}_0 + \mathbf{v}_1D + \mathbf{v}_2D^2 + \dots$$

Algebraic description



$$\mathbf{v}(D) = \mathbf{u}(D) \frac{f(D)}{q(D)} = \mathbf{u}(D) \frac{f_0 + f_1 D + f_m D^m}{1 + q_1 D + \dots + q_m D^m}.$$

Definition

Let

$$\mathbb{F}[D] = \left\{ \sum_{i=0}^d x_i D^i : x_i \in \mathbb{F}, 0 \leq i \leq d < \infty \right\}$$

be the set of binary formal polynomials.

Definition

Let

$$\mathbb{F}(D) = \left\{ \frac{p(D)}{q(D)} : p(D), q(D) \in \mathbb{F}[D] \right\}$$

be the set of binary rational functions.

Definition

Let

$$\mathbb{F}[[D]] = \left\{ \sum_{i=0}^{\infty} x_i D^i : x_i \in \mathbb{F}, 0 \leq i \right\}$$

be the set of formal power series.

Definition

Let

$$\mathbb{F}((D)) = \left\{ \sum_{i=-\infty}^{\infty} x_i D^i : x_i \in \mathbb{F}, i \in \mathbb{Z} \right\}$$

be the set of binary Laurent series.

Main definitions

Definition

Given $x(D) \in \mathbb{F}((D))$, its delay $\text{del}(x(D))$ is the smallest r for which $x_r = 1$.

Definition

Given $x(D)$ is called delay free if $\text{del}(x(D)) = 0$.

Definition

A rational function is called realizable if $q(D)$ is delay free.

A rate $R = b/c$ binary convolutional encoder is a linear mapping

$$\Phi : \mathbb{F}^b((D)) \rightarrow \mathbb{F}^c((D)),$$

which can be represented as

$$\mathbf{v}(D) = \mathbf{u}(D)G(D),$$

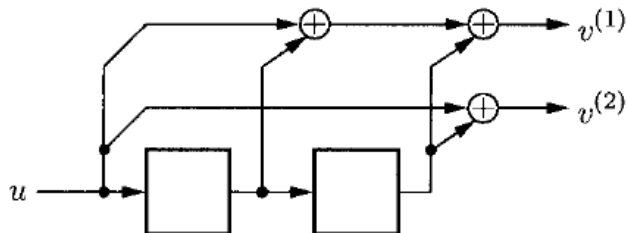
where $G(D)$ is of size $b \times c$ with entries from $\mathbb{F}(D)$.

Definition

$G(D)$ is a polynomial generator matrix if all the elements are realizable.

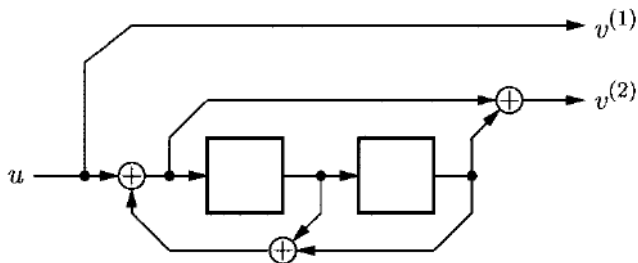
Example

$$G(D) = (1 + D + D^2, 1 + D^2)$$



Systematic encoder

$$G(D) = (1, \frac{1 + D^2}{1 + D + D^2})$$



- 1 Soft decoding
- 2 First look at convolutional codes
- 3 Termination methods
- 4 Algebraic description
- 5 Distance properties**
- 6 Decoding

Free distance

$$d_f = \min_{\gamma} ||\gamma||,$$

where γ is a path on trellis with initial and final states equal to q_0 .

- 1 Soft decoding
- 2 First look at convolutional codes
- 3 Termination methods
- 4 Algebraic description
- 5 Distance properties
- 6 Decoding

Bit-wise MAP vs Block-wise MAP decoding

Block-wise MAP decoding

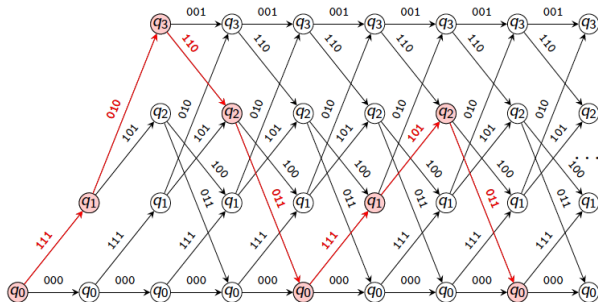
$$\hat{x}^{(MAP)}(y) = \arg \max_{x \in \mathcal{C}} \Pr(x|y).$$

Bit-wise MAP decoding

$$\hat{x}_i^{(MAP)}(y) = \arg \max_{x_i \in \{+1, -1\}} \sum_{x \in \mathcal{C}, x_i \text{ is fixed}} \Pr(x|y).$$

Viterbi algorithm

Again consider a trellis. Assume we use a zero-termination of length n , so we should only consider paths which start in state q_0 and end in q_0 (one of such paths is shown below)



1100100... \mapsto 111 010 110 011 111 101 011...

Assume we send a codeword $\mathbf{x} \in \mathbb{F}_2^n$ and received a sequence \mathbf{y} .

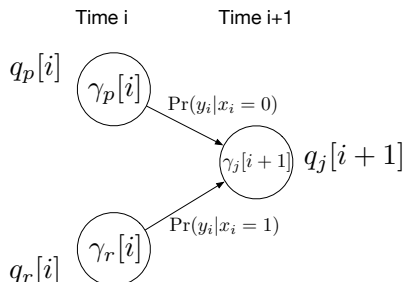
Recall, that the channel is memoryless, then

$$\Pr(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^n \Pr(y_i|x_i).$$

Label the edges of trellis with $\Pr(y_i|x_i)$ values.

Dynamic programming rule

Viterbi algorithm is an instance of dynamic programming. Consider trellis states at time $i + 1$.



$$\gamma_j[i+1] = \max\{\gamma_p[i] \Pr(y_i|x_i = 0), \gamma_r[i] \Pr(y_i|x_i = 1)\}$$

Thank you for your attention!