Background

Convolutional codes do not segment the data stream, but convert the entire data stream into another stream (codeword). Some facts about convolutional codes:

- Introduced by Elias in 1955.
- Constructed mainly using heuristic techniques (cf. block codes: algebraic and combinatorial techniques).
- Can be decoded in an "asymptotically optimal" way using the so-called Viterbi algorithm (which Forney proved to be a maximum-likelihood decoding algorithm for convolutional codes).

Linear Convolutional Encoders (1)

A typical rate-1/2 linear convolutional encoder is shown in [Wic, Fig. 11-1]. An encoder with k inputs and n outputs is said to have rate k/n. A rate-2/3 encoder is shown in [Wic, Fig. 11-2].

The k input and n output streams are denoted by $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(k-1)}$, and $\mathbf{y}^{(0)}, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n-1)}$, respectively. The data of an input stream is denoted by $\mathbf{x}^{(i)} = (x_0^{(i)}, x_1^{(i)}, \ldots)$, and analogously for $\mathbf{y}^{(i)}$.

From multiple output streams, we can create a single (interleaved) output stream $\mathbf{y} = (y_0^{(0)}, y_0^{(1)}, \ldots, y_0^{(n-1)}, y_1^{(0)}, \ldots)$, and an analogous notation can be used for multiple inputs.

Linear Convolutional Encoders (2)

Linear convolutional encoders can be viewed as, for example,

- finite impulse response (FIR) digital filters, or
- finite state automata.

It is known that every linear convolutional encoder is equivalent to a *minimal encoder* that is *feedback-free*.

Output as Function of Input (1)

Each element in the interleaved output stream $\mathbf{y}$ is a combination of the elements in the input stream $\mathbf{x}$. (Note: The shift-register contents are initialized to zero before the encoding process begins.)

Example. In the encoder in [Wic, Fig. 11-1], we have

$$\begin{aligned} y_0^{(1)} &= x_0^{(0)} + 0 + 0, \\ y_1^{(1)} &= x_1^{(0)} + x_0^{(0)} + 0, \\ y_2^{(1)} &= x_2^{(0)} + x_1^{(0)} + 0, \\ y_3^{(1)} &= x_3^{(0)} + x_2^{(0)} + x_0^{(0)}, \\ &\vdots \\ y_i^{(1)} &= x_i^{(0)} + x_{i-1}^{(0)} + x_{i-3}^{(0)}. \end{aligned}$$

Output as Function of Input (2)

Example. (cont.) From the last expression on the previous page, it is clear that if $\mathbf{y}'$ and $\mathbf{y}''$ are the codewords corresponding to inputs $\mathbf{x}'$ and $\mathbf{x}''$, respectively, then $\mathbf{y}' + \mathbf{y}''$ is the codeword corresponding to the input $\mathbf{x}' + \mathbf{x}''$, so the code is *linear*. In this example, if $\mathbf{x}^{(0)} = (10110)$, then $\mathbf{y}^{(1)} = (11111110)$.

The linear structure of these codes allows for the use of some powerful techniques from linear systems theory.

Impulse Response

An **impulse response** $\mathbf{g}_j^{(i)}$ is obtained for the ith output of an encoder by applying a single 1 at the jth input followed by a string of zeros: $\mathbf{x}^{(j)} = \delta = (1000\ldots)$. Strings of zeros are applied to all other inputs. The impulse response is terminated at the point from which the output contains only zeros.

Example. For the encoder in [Wic, Fig. 11-1], we have

$$\begin{aligned} \mathbf{g}_0^{(0)} &= (1011), \\ \mathbf{g}_0^{(1)} &= (1101). \end{aligned}$$

(Notation: $g_{j,l}^{(i)}$ is bit l of $\mathbf{g}_j^{(i)}$.) The impulse responses are often referred to as **generator sequences**.

Constraint Length

The **constraint length** K of a convolutional code is the maximum number of bits in a single output stream that can be affected by any input bit.

In practice, another definition is often used: The constraint length is the length of the longest input shift register plus one:

$$K := 1 + m,$$

where $m := \max_i m_i$ is called the *maximal memory order* and m_i is the length of the shift register into which $\mathbf{x}^{(i)}$ is fed. The *total memory* for a convolutional encoder is defined as $\sum_{i=0}^{k-1} m_i$.

Fractional Rate Loss

Asymptotically, the ratio between the number of input bits and output bits tends to $R = k/n$. For an input stream of length L, the ratio is not exactly R, but we have a *fractional rate loss*

$$\gamma = \frac{R - R_{\text{eff}}}{R} = \left(\frac{k}{n}\right)^{-1} \left\{ \left(\frac{k}{n}\right) - \left[\frac{L}{\left(\frac{n}{k}\right) L + nm} \right] \right\} = \frac{km}{L + km}.$$

Example. For the encoder in [Wic, Fig. 11-1] we have $k = 1$ and $m = 3$, so with an input of length $L = 5$ (as in an earlier example), we get

$$\gamma = \frac{km}{L + km} = \frac{1 \cdot 3}{5 + 1 \cdot 3} = 0.375.$$

Convolutions of Sequences

An output stream can be expressed as a function of the input streams and the generator sequences

$$y_j^{(i)} = \sum_{t=0}^{k-1} \left(\sum_{l=0}^{m} x_{j-l}^{(t)} g_{t,l}^{(i)} \right),$$

which is a sum of discrete *convolutions* of pairs of sequences:

$$\mathbf{y}^{(i)} = \sum_{t=0}^{k-1} \mathbf{x}^{(t)} * \mathbf{g}_t^{(i)}.$$

Convolutional Generator Matrix

The previous expressions can be re-expressed as a matrix multiplication operation, thus obtaining a semi-infinite generator matrix.

For a rate-1/2 code, the generator matrix is formed by interleaving the two generator sequences as follows:

$$\mathbf{G} = \begin{bmatrix} g_0^{(0)} & g_0^{(1)} & g_1^{(0)} & g_1^{(1)} & \cdots & g_m^{(0)} & g_m^{(1)} & & \\ & & g_0^{(0)} & g_0^{(1)} & g_1^{(0)} & g_1^{(1)} & \cdots & g_m^{(0)} & g_m^{(1)} \\ & & & \ddots & & \ddots & & \ddots & \end{bmatrix}.$$

Example: Convolutional Generator Matrix

The sequence $\mathbf{x} = (1011)$ is to be encoded using the rate-1/2 encoder in [Wic, Fig. 11-1]. For this code, $\mathbf{g}^{(0)} = (1011)$ and $\mathbf{g}^{(1)} = (1101)$, so

$$\mathbf{y} = \mathbf{xG} = (1011) \begin{bmatrix} 11 & 01 & 10 & 11 & 00 & 00 & 00 \\ 00 & 11 & 01 & 10 & 11 & 00 & 00 \\ 00 & 00 & 11 & 01 & 10 & 11 & 00 \\ 00 & 00 & 00 & 11 & 01 & 10 & 11 \end{bmatrix} =$$

$(11, 01, 01, 01, 11, 01, 11)$.

The Delay Transform (1)

An appropriate transform will provide a simpler multiplicative representation for encoding. In this case, we apply the **delay transform** (also called the D-transform):

$$\begin{aligned} \mathbf{x}^{(i)} = (x_0^{(i)}, x_1^{(i)}, \ldots) &\quad \leftrightarrow \quad \mathbf{X}^{(i)}(D) = x_0^{(i)} + x_1^{(i)} D + \cdots, \\ \mathbf{y}^{(i)} = (y_0^{(i)}, y_1^{(i)}, \ldots) &\quad \leftrightarrow \quad \mathbf{Y}^{(i)}(D) = y_0^{(i)} + y_1^{(i)} D + \cdots, \\ \mathbf{g}_j^{(i)} = (g_{j,0}^{(i)}, g_{j,1}^{(i)}, \ldots) &\quad \leftrightarrow \quad \mathbf{G}_j^{(i)}(D) = g_{j,0}^{(i)} + g_{j,1}^{(i)} D + \cdots. \end{aligned}$$

The Delay Transform (2)

The encoding operation can now be represented as

$$\mathbf{Y}^{(i)}(D) = \sum_{j=0}^{k-1} \mathbf{X}^{(j)}(D)\mathbf{G}_j^{(i)}(D),$$

or as (the matrix $\mathbf{G}(D)$ is called a *transfer-function matrix*)

$$\mathbf{Y}(D) = \mathbf{X}(D)\mathbf{G}(D) = (\mathbf{X}^{(0)}(D)\ \mathbf{X}^{(1)}(D)\ \cdots\ \mathbf{X}^{(k-1)}(D))\cdot$$

$$\begin{bmatrix} \mathbf{G}_0^{(0)}(D) & \mathbf{G}_0^{(1)}(D) & \cdots & \mathbf{G}_0^{(n-1)}(D) \\ \mathbf{G}_1^{(0)}(D) & \mathbf{G}_1^{(1)}(D) & \cdots & \mathbf{G}_1^{(n-1)}(D) \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{G}_{k-1}^{(0)}(D) & \mathbf{G}_{k-1}^{(1)}(D) & \cdots & \mathbf{G}_{k-1}^{(n-1)}(D) \end{bmatrix}.$$

Example: The Delay Transform

The rate-2/3 encoder in [Wic, Fig. 11-2] is used to encode the message $x = (11, 10, 11)$. Then $\mathbf{X}^{(0)}(D) = 1 + D + D^2$ and $\mathbf{X}^{(1)}(D) = 1 + D^2$, so

$$\mathbf{Y}(D) = (1+D+D^2\ 1+D^2) \begin{bmatrix} 1+D^3 & D+D^2+D^3 & 1+D \\ D+D^2 & 1+D^2 & D \end{bmatrix} =$$

$$(1+D^5\ 1+D+D^3+D^4+D^5\ 1+D).$$

Hence $\mathbf{y}^{(0)} = (100001)$, $\mathbf{y}^{(1)} = (110111)$, and $\mathbf{y}^{(2)} = (110000)$, and the output word is $\mathbf{y} = (111, 011, 000, 010, 010, 110)$.

Systematic Convolutional Codes

A convolutional code is said to be **systematic** if the input data is reproduced unaltered in the output codeword.

A rate-1/2 systematic convolutional encoder is shown in [Wic, Fig. 11-3]. In a rate-k/n systematic encoder, k of the n output coded data streams are identical to the k input streams. The matrix $\mathbf{G}(D)$ of a systematic encoder contains the $k \times k$ identity matrix $\mathbf{I}_k$.

Note: Not every convolutional code has a systematic encoding (which is the case for block codes). In fact, many good convolutional codes are not systematic.

Properties of Convolutional Codes

Convolutional codes differ from block codes in several ways, some mentioned earlier. Most of the techniques used for analyzing and comparing block codes cannot be applied to convolutional codes. For example, what about minimum distance?

There are two important graphical techniques for analyzing convolutional codes:

1. State diagrams.
2. Trellis diagrams.

State Diagrams (1)

A convolutional encoder is a finite-state automaton, where the next output depends only on the input bits and the contents of its memory cells. With k memory cells, the finite-state automaton has 2^k states.

Example. The encoder in [Wic, Fig. 11-1] has 8 states, which can be associated with the memory contents as follows:

$$S_0 \leftrightarrow (000), \quad S_4 \leftrightarrow (001),$$
$$S_1 \leftrightarrow (100), \quad S_5 \leftrightarrow (101),$$
$$S_2 \leftrightarrow (010), \quad S_6 \leftrightarrow (011),$$
$$S_3 \leftrightarrow (110), \quad S_7 \leftrightarrow (111).$$

State Diagrams (2)

Example. (cont.) The **state diagram** of this encoder is shown in [Wic, Fig. 11-4]. Each branch in the state diagram has a label of the for X/YY, where X is the input bit that causes the state transition and YY is the corresponding pair of output bits.

Some Graph Concepts

graph Consists of vertices (nodes) and edges (branches) that connect the vertices.

directed graph A graph where the edges have an associated direction.

weighted graph A graph where (usually nonnegative integer) values are associated with the edges.

path A sequence of vertices in which consecutive vertices are connected with an edge (in the correct direction if the graph is directed).

circuit A path that starts and stops at the same vertex.

cycle A circuit that does not enter any vertex more than once.

State Diagrams (3)

We may consider the state diagram as a *weighted directed graph*, where the weight of an edge is the Hamming weight of the output bits. State diagrams are therefore also called *encoder graphs*.

The encoding process begins and ends in the all-zero state, so every convolutional codeword is associated with a circuit through the encoder graph that starts and stops at state S_0.

Catastrophic Convolutional Codes

A convolutional code is said to be **catastrophic** if its corresponding state diagram contains a cycle in which a nonzero input sequence corresponds to an all-zero output sequence.

With a catastrophic code, a small number of channel errors can cause an unlimited number of errors in the decoded data stream.

Example: Catastrophic Code

The encoder graph of a catastrophic code is shown in [Wic, Fig. 11-5] (and the corresponding encoder is depicted in [Wic, Fig. 11-6]). The loop about state S_7 shows that the code is catastrophic. With this encoder, the all-zero word $(0, 0, \ldots, 0)$ is encoded as

$$(00, 00, \ldots, 00),$$

and the all-one word $(1, 1, \ldots, 1)$ is encoded as

$$(11, 00, 11, 00, 00, \ldots, 00).$$

Then there are situations with a finite (small) number of channel errors where maximum-likelihood decoding will lead to an infinite number of errors.

Avoiding Catastrophic Codes

To avoid the weight-zero loop about the all-ones state, one need only assure that at least one of the output streams is formed through the summation of an odd number of terms. This condition is not sufficient, however, as shown by the encoder and the encoder graph given in [Wic, Fig. 11-8] and [Wic, Fig. 11-7], respectively.

Fortunately, catastrophic codes are relatively infrequent (for example, only $1/(2^n - 1)$ of all convolutional codes of rate $1/n$ and a given constraint length are catastrophic).

A set of necessary and sufficient conditions for a convolutional code to be noncatastrophic has been proved by Massey and Sain.

Conditions for Catastrophic Codes

1. Let C be a rate-$1/n$ convolutional code with transfer-function matrix $\mathbf{G}(D)$ whose generator sequences have the transforms $\mathbf{G}^{(i)}(D)$, $0 \leq i \leq n-1$. Then C is not catastrophic iff for some nonnegative integer l,

$$\mathrm{GCD}(\mathbf{G}^{(0)}(D), \mathbf{G}^{(1)}(D), \ldots, \mathbf{G}^{(n-1)}(D)) = D^l.$$

2. Let C be a rate-k/n convolutional code with transfer-function matrix $\mathbf{G}(D)$. Then C is not catastrophic iff for some nonnegative integer l,

$$GCD(\Delta_i(D), i \in S) = D^l,$$

where $\Delta_i(D)$ is the determinant of the ith $k \times k$ submatrix of $\mathbf{G}(D)$ and S contains the indexes of all such submatrices.

Example: Conditions for Convolutional Codes

The generator sequences for the encoder in [Wic, Fig. 11-8] are

$$\mathbf{g}^{(0)} = (0111),$$
$$\mathbf{g}^{(1)} = (1110),$$

and the corresponding D-transforms are

$$\mathbf{G}^{(0)} = D + D^2 + D^3,$$
$$\mathbf{G}^{(1)} = 1 + D + D^2.$$

Then $\mathrm{GCD}(\mathbf{G}^{(0)}, \mathbf{G}^{(1)}) = 1 + D + D^2$, and since $1 + D + D^2 \neq D^l$ for any integer l, the code is catastrophic.