Recitation 9

Last Time:
- Encoding CC’s using various methods
  - state table, state diagram, generator sequence, D-transform, etc.

Today:
- Viterbi Decoding of CC’s.

Example 1:
* Consider the convolutional encoder defined by $g_1(x) = (110)$ and $g_2(x) = (110)$ shown below.

The corresponding state diagram is as below.

We can construct a trellis diagram, very useful in decoding a received vector $\tilde{y}$.

Trellis: an extension of state diagram with explicit time passage!

Observations:
1. As with state diagram, trellis has $2^m = 2^{k-1}$ states; for $m$-memory or constraint length $n$ encoder.
2. For $(n, k)$ binary CC, each node has $2^k$ branches leaving and $2^k$ branches entering.
3. We’ll assume that encoder starts at zero state (state $s_0$) that encoder is time-invariant, so $\tilde{y}$ is constant.
   - that noise is uncorrelated from stage to stage.
4. After all bits of input vector \( x \) entered, need \( m \)-state transitions to return encoder to all-zero state \( S_0 \). Thus, given input sequence length \( x \), trellis must have \( 1 + m \) stages, starting and stopping at \( S_0 \).

5. There are \( 2^m \) distinct paths through trellis, each corresponding to a length \( nL + m \) possible codeword.

(We'll see how Viterbi addresses this issue)

Below is the trellis diagram for the encoder in our example. (Copied from lecture notes)

---

**The Decoding Problem**

Recall our baseband model:

\[
\begin{array}{c}
\times \\
\downarrow \quad \text{Encoder} \\
\downarrow \\
\downarrow \quad \text{Channel} \\
\downarrow \\
\downarrow \quad \text{Noise} \\
\downarrow \\
\downarrow \quad \text{Decoder} \\
\Rightarrow \\
\Rightarrow \quad \hat{r} \\
\Rightarrow \\
\Rightarrow \quad \theta \quad \text{ML decoder: selects estimate that maximizes } P(\hat{y} | \frac{1}{2})
\end{array}
\]

Naïve decoder: compares \( \hat{r} \) with \( 2^m \) possible 1-bit sequences (e.g., for 16-bit \( \hat{r} \), need \( 64K \) comparisons).

Viterbi decoder: Need not consider all \( 2^m \) possible paths in trellis; at any stage, only \( 2^m \) paths need be retained.

Assumes: channel is memoryless (i.e., errors uncorrelated from stage to stage).

Theorem: The path selected by the Viterbi decoder is the ML path! (Linear vs. O(exponential))
The Viterbi Algorithm:

Define:
$S_j,t$ = node $S_j$ at time $t$.

$PM(S_j,t)$ = path metric of node $S_j$ at time $t$.

VA:
1. Initialize $t = 0$
   Set $PM(S_{0,0}) = 0$; $PM(S_{j,0}) = \infty \forall j \neq 0$.
2. While $(t < L+m)$
   increment $t$.
   Compute PPMs (partial path metrics) of paths entering each node.
   Set $PM(S_{j,t})$ = Best PPM entering node $S_j$ at time $t$.
   Delete non-surviving branches.
3. Trace back from $S_0$ at time $t = L+m$, following surviving branches.
   The path thus defined is the unique ML codeword!

Example: Our example encoder encodes the message sequence $\mathbf{x} = (11001)$, generating the codeword $\mathbf{y} = (11, 00, 01, 10, 11, 11, 10)$. If $\mathbf{y}$ is transmitted over a noisy BSC, so that the received word is $\mathbf{r} = (11, 00, 11, 10, 12, 11, 10)$. Use the Viterbi decoder to obtain the ML codeword $\hat{\mathbf{x}}$ and corresponding transmitted sequence $\hat{\mathbf{y}}$.

Solv: We'll use the constructed trellis with calculated PPMs, BMs and PMs using HD (d_H) as our metric (next page).

We obtain $\hat{\mathbf{x}} = (11, 00, 01, 10, 11, 11, 10)$ which corrected the errors!

The transmitted estimated sequence is thus (11001) as expected.
ML path minimizes $d_H(\hat{x}, \hat{y})$, so choose $\min PM$.

$$PM(s_{ij}, \hat{y}) = \min \left\{ PM(s_{ij}, \hat{y}) + BM(s_{ij}, s_{ij}), s_{ij} \text{ branches into } s_{ij} \right\}$$

where $BM = d_H(\hat{x}, \hat{y})$ at stage.