

13 Appendix A: Hidden Markov Sources

13.1 MAP Algorithm to Estimate Hidden Markov Source

- Encoder generates \mathbf{x} from message \mathbf{m} .
- \mathbf{x} is channel input, \mathbf{y} is channel output.

Discrete memoryless channel (DMC)

1. Inputs are from a finite set.
2. Individual transmissions are statistically independent.

Point 2 requires that the conditional output distribution be expressed in product form:

$$P(\mathbf{y}|\mathbf{x}) = \prod_{j=1}^n R_j(y_j|x_j)$$

Definition 1 The channel is **stationary** if R_j does not change with j . □

Encoder output

The **source distribution** for the j^{th} transmission is given by

$$P(x_j) \triangleq P[x = x_j]$$

and is called the **source marginal probability** distribution.

The usual sort of **uncoded** source output is given by:

$$P(\mathbf{x}) = P(x_1, x_2, \dots, x_n) = \prod_{j=1}^n P(x_j)$$

but this does not account for dependencies in the output from a Finite State Machine.

So we model the channel input as a **Discrete Markov Source (DMS)**:

1. the symbols are drawn from a finite set;
2. The joint probability of the input block is given by:

$$P(\mathbf{X}) = P(x_1, x_2, \dots, x_n) = P(x_1) \cdot \prod_{j=1}^n Q_j(x_j | x_{j-1}) \quad (2)$$

Use of Bayes's Rule shows the dependence of one source output symbol upon the next:

$$Q_j(x_j | x_{j-1}) = \frac{P(x_{j-1}, x_j)}{P(x_{j-1})}$$

And the source marginals become

$$P(x_j) = \sum_{x_{j-1} \in \mathbb{X}_{j-1}} P_{j-1}(x_{j-1}) \cdot Q_j(x_j | x_{j-1}),$$

where X_{j-1} takes values on the alphabet \mathbb{X}_{j-1} .

The DMS is stationary if the transition probabilities are independent of j .

Bayes's Rule also gives the **source reverse transition probabilities**:

$$\tilde{Q} = \frac{P(x_{j-1}, x_j)}{P(x_j)} = \begin{cases} \frac{P(x_{j-1})Q_j(x_j|x_{j-1})}{P_j(x_j)} & P_j(x_j) > 0 \\ 0 & \text{otherwise} \end{cases}$$

and this leads to another expression for the source block probability:

$$P(\mathbf{x}) = P_n(x_n) \cdot \prod_{j=n}^2 \tilde{Q}_j(x_{j-1}|x_j) \quad (3)$$

Compare (3) with (2).

Past, Present, and Future

At any time j :

- x_j = the present source symbol;
- the **past** is $\mathbf{x}_j^- = (x_1, x_2, \dots, x_{j-1})$;
- the **future** is $\mathbf{x}_j^+ = (x_{j+1}, x_{j+2}, \dots, x_n)$;
- the past and future are **independent**:

i.e., we can write the joint probability as:

$$\begin{aligned} p(\mathbf{x}) &= p(\mathbf{x}_j^-, x_j, \mathbf{x}_j^+) \\ &= P(\mathbf{x}_j^- | x_j) \cdot P(x_j) \cdot P(\mathbf{x}_j^+ | x_j) \end{aligned} \quad (4)$$

In general

- the actual information source may or may not be Markov;
- in any case, the **encoder output** is a DMS;
- if the **information source transition probabilities** are not known, then the encoder output is a **discrete hidden Markov source (DHMS)**.

Definition 2 A **discrete hidden Markov source (DHMS)** is obtained whenever a memoryless function (such as a DMC) is applied to a DMS. \square

i.e.,

- the source is “hidden” by the DMC; we can observe only the output;
- we must *determine the input \mathbf{x} , a DHMS, from the output \mathbf{y} of a DMC*;
- the method uses a MAP algorithm to solve for x_j recursively.

Set up the MAP algorithm

Since $\mu_{MAP}(x) = P(x|y) = \frac{P(x,y)}{P(y)}$,

$$\mu_{MAP}(x) \propto P(x, y)$$

So, the MAP decoder requires the joint distribution of \mathbf{y} , and x_j at time j .

From (4):

$$P(x_j, \mathbf{y}) = P(x_j, \mathbf{y}_j^-) \cdot P(y_j | x_j, \mathbf{y}_j^-) \cdot P(\mathbf{y}_j^+ | x_j, y_j, \mathbf{y}_j^-)$$

But \mathbf{y} is a DMS, so

1. the middle factor is the DMC, so is independent of prior history (\mathbf{y}_j^-);
2. \mathbf{y}_j^+ is independent of y_j and \mathbf{y}_j^- .

So the joint probability is

$$P(x_j, \mathbf{y}) = P(x_j, \mathbf{y}_j^-) \cdot P(y_j | x_j) \cdot P(\mathbf{y}_j^+ | x_j) \quad (5)$$

The three factors represent past, current, and present values of sequence \mathbf{y} .

It is customary to write:

$$\begin{aligned}\alpha_j(x_j) &\triangleq P(x_j, \mathbf{y}_j^-) \\ \beta_j(x_j) &\triangleq P(\mathbf{y}_j^+ | x_j) \\ \gamma_j(x_j) &\triangleq P(y_j | x_j) = R_j(y_j | x_j)\end{aligned}$$

Substitution into (5) gives the **generic MAP equation for a DMS over a DMC**

$$P(x_j, \mathbf{y}) = \alpha_j(x_j) \cdot \gamma_j(x_j) \cdot \beta_j(x_j) \quad (6)$$

Recursions

$$\begin{aligned}
 \alpha_j(x_j) &\triangleq P(x_j, \mathbf{y}_j^-) \\
 &= \sum_{x_{j-1} \in \mathbb{X}_{j-1}} P(x_{j-1}, x_j, \mathbf{y}_j^-) \\
 &= \sum_{x_{j-1} \in \mathbb{X}_{j-1}} \underbrace{P(x_{j-1}, \mathbf{y}_{j-1}^-) \cdot P(x_j | x_{j-1}, \mathbf{y}_{j-1}^-)}_{P(x_{j-1}, \mathbf{y}_{j-1}^-, x_j)} \cdot P(y_{j-1} | x_{j-1}, x_j, \mathbf{y}_{j-1}^-) \\
 &= \sum_{x_{j-1} \in \mathbb{X}_{j-1}} P(x_{j-1}, \mathbf{y}_{j-1}^-) \cdot P(x_j | x_{j-1}) \cdot P(y_{j-1} | x_{j-1}) \\
 &= \sum_{x_{j-1} \in \mathbb{X}_{j-1}} \alpha_{j-1} s_{-j-1} Q_j(x_j | x_{j-1}) R_j(y_{j-1} | x_{j-1}).
 \end{aligned}$$

The 2nd and 3rd factors do not depend on \mathbf{y}_{j-1}^- ; the channel is a DMC.
Initial cond.: $\alpha_1(x_1) = P(x_1)$.

We will need a **backward recursion** for β :

$$\begin{aligned}
 \beta_j(x_j) &\triangleq P(\mathbf{y}_j^+ | x_j) \\
 &= \sum_{x_{j+1} \in \mathbb{X}_{j+1}} P(x_{j+1}, \mathbf{y}_j^+ | x_j) \\
 &= \sum_{x_{j+1} \in \mathbb{X}_{j+1}} P(x_{j+1} | x_j) \cdot P(y_{j+1} | x_j, x_{j+1}) \cdot P(\mathbf{y}_{j+1}^+ | x_j, x_{j+1}, y_{j+1}) \\
 &= \sum_{x_{j+1} \in \mathbb{X}_{j+1}} P(x_{j+1} | x_j) \cdot P(y_{j+1} | x_{j+1}) \cdot P(\mathbf{y}_{j+1}^+, x_{j+1}) \\
 &= \sum_{x_{j+1} \in \mathbb{X}_{j+1}} Q_{j+1}(x_{j+1} | x_j) \cdot R_{j+1}(y_{j+1} | x_{j+1}) \beta_{j+1}(x_{j+1})
 \end{aligned}$$

Initial cond.: $\beta_n(x_n) = 1$ if $P(x_n) > 0$ and $= 0$ otherwise.

General MAP Analysis: Binary case

- Let i_ℓ be the ℓ^{th} info bit;
- a block of information is $\mathbf{i} = (i_0, i_1, \dots, i_{k-1})$;
- actually want the MAP estimate of the information:

$$\mu_{MAP}(i_\ell) = \ln \frac{P(i_\ell = +1|\mathbf{y})}{P(i_\ell = -1|\mathbf{y})}$$

Bayes gives:

$$P(i_\ell = +1|\mathbf{y}) = \frac{P(i_\ell = +1, \mathbf{y})}{P(\mathbf{y})}.$$

For each information sequence $\mathbf{i} = (i_0, i_1, \dots, i_{k-1})$, there is exactly one code sequence $\mathbf{x} = (x_0, x_1, \dots, x_{n-1})$. So, $P(\mathbf{y}) = \sum_{\mathbf{x}} P(\mathbf{y}|\mathbf{x})P(\mathbf{x})$ implies:

$$P(\mathbf{y}) = \sum_{\mathbf{i}} P(\mathbf{y}|\mathbf{x})P(\mathbf{i})$$

i.e., we understand \mathbf{x} to be the codeword corresponding to info sequence \mathbf{i} .

Let $\mathbf{I}_\ell^+ = \{\text{all information seqs } \mathbf{i} : i_\ell = +1\}$ and $\mathbf{I}_\ell^- = \{\mathbf{i} : i_\ell = -1\}$. Then,

$$P(i_\ell = +1, \mathbf{y}) = \sum_{\mathbf{i} \in \mathbf{I}_\ell^+} P(\mathbf{y}|\mathbf{x}) \cdot P(\mathbf{i})$$

Similarly

$$P(i_\ell = -1, \mathbf{y}) = \sum_{\mathbf{i} \in \mathbf{I}_\ell^-} P(\mathbf{y}|\mathbf{x}) \cdot P(\mathbf{i})$$

Combining the last three equations gives:

$$\mu_{MAP}(i_\ell) = \ln \frac{\sum_{\mathbf{i} \in \mathbf{I}^+} P(\mathbf{y}|\mathbf{x}) \cdot P(\mathbf{i})}{\sum_{\mathbf{i} \in \mathbf{I}^-} P(\mathbf{y}|\mathbf{x}) \cdot P(\mathbf{i})}, \quad (7)$$

from which μ_{MAP} can be computed directly for every possible value of i_ℓ , and

$$\hat{i}_\ell = \begin{cases} +1, & \mu_{MAP}(i_\ell) > 0 \\ -1, & \mu_{MAP}(i_\ell) < 0 \end{cases}, \quad \ell = 0, 1, \dots, k-1.$$

But,

- to estimate (7) is a brute force calculation.
- It works for any DHMS, even one with no “regular” structure.
- It does **not** exploit the structure of a source that is defined on a trellis.
 - The foregoing general method iterates over every branch in the trellis.
 - It searches over $2^m \cdot 2^k$ branches in the trellis.
- It is possible to search over the 2^m **states** of the trellis to get the MAP estimate;
- measures $\alpha_j(s_j)$, $\beta_j(s_j)$, etc on the states of the trellis will be defined.

13.2 Algorithm for Estimating Trellis Source over DMC

Comparison

DMS over DMC	Trellis source over DMC
$P(x_j, \mathbf{y}) = \alpha_j(x_j)\beta_j(x_j)\gamma_j(x_j)$	$P(i_\ell, \mathbf{y}) = \alpha_{\ell-1}(s_{\ell-1})\gamma_\ell(i_\ell)\beta_\ell(s_\ell)$
$\mu_{MAP}(x) \propto P(x, \mathbf{y})$	$\mu_{MAP}(i) \propto P(i, \mathbf{y})$

Using a derivation similar to that for the general DMS over a DMC, we also define:

$$s_{\ell-1} = \sigma^-(i_\ell) \text{ (state at beginning of branch)}$$

$$s_\ell = \sigma^+(i_\ell) \text{ (state at end of branch)}$$

MAP Trellis Decoder for DMC

The **log likelihood ratio** for the ℓ^{th} information bit:

$$L(i_\ell) \triangleq \ln \frac{P(i_\ell = +1|\mathbf{y})}{P(i_\ell = -1|\mathbf{y})} \quad (8)$$

Let \mathbf{I}_ℓ^+ = the set of all (binary) information sequences in which $i_\ell = +1$; let \mathbf{I}_ℓ^- be analogously defined. Then, from Bayes,

$$\begin{aligned} P(i_\ell = +1|\mathbf{y}) &= \frac{P(i_\ell = +1, \mathbf{y})}{P(\mathbf{y})} \\ &= \frac{\sum_{\mathbf{i} \in \mathbf{I}_\ell^+} P(\mathbf{y}|\mathbf{x})P(\mathbf{i})}{P(\mathbf{y})} \end{aligned}$$

Substitution into (8) gives

$$L(\mathbf{i}_\ell) = \ln \frac{\sum_{\mathbf{i} \in \mathbf{I}_\ell^+} P(\mathbf{y}|\mathbf{x})P(\mathbf{i})}{\sum_{\mathbf{i} \in \mathbf{I}_\ell^-} P(\mathbf{y}|\mathbf{x})P(\mathbf{i})} \quad (9)$$

Now, the DHMS over DMC problem can be re-formulated on a trellis:

Lemma 1 *Let Σ_ℓ^+ = the set of state pairs $\{(s_\ell, s_{\ell+1}) : i_\ell = +1\}$ where the state transition $s_\ell \rightarrow s_{\ell+1}$ is determined by the information bit i_ℓ . Let Σ_ℓ^- be analogously defined. Then*

$$P(i_\ell = +1, \mathbf{y}) = \sum_{s_\ell, s_{\ell+1} \in \Sigma_\ell^+} P(s_\ell, s_{\ell+1}, \mathbf{y})$$

$$P(i_\ell = -1, \mathbf{y}) = \sum_{s_\ell, s_{\ell+1} \in \Sigma_\ell^-} P(s_\ell, s_{\ell+1}, \mathbf{y})$$

Proof: by reasoning on the trellis structure. □

So the results of the lemma are substituted into (8), giving:

$$L(i_\ell) = \ln \frac{\sum_{s_\ell, s_{\ell+1} \in \Sigma_\ell^+} P(s_\ell, s_{\ell+1}, \mathbf{y})}{\sum_{s_\ell, s_{\ell+1} \in \Sigma_\ell^-} P(s_\ell, s_{\ell+1}, \mathbf{y})} \quad (10)$$

- The summation of (10) is less complex than summing over all information sequences, and every branch, $s_\ell \rightarrow s_{\ell+1}$ belongs to either Σ_ℓ^+ or Σ_ℓ^- .
- The current output \mathbf{y}_ℓ is a now vector of length n the branch output.

Now, expand the joint probability of states and output:

$$\begin{aligned}
 P(s_\ell, s_{\ell+1}, \mathbf{y}) &= P(s_\ell, s_{\ell+1}, \mathbf{y}_\ell^-, \mathbf{y}_\ell, \mathbf{y}_\ell^+) \\
 &= P(\mathbf{y}_\ell^+ | s_\ell, s_{\ell+1}, \mathbf{y}_\ell^-, \mathbf{y}_\ell) \cdot P(s_\ell, s_{\ell+1}, \mathbf{y}_\ell^-, \mathbf{y}_\ell) \\
 &= P(\mathbf{y}_\ell^+ | s_\ell, s_{\ell+1}, \mathbf{y}_\ell^-, \mathbf{y}_\ell) \cdot P(s_{\ell+1}, \mathbf{y}_\ell | s_\ell, \mathbf{y}_\ell^-) \cdot P(s_\ell, \mathbf{y}_\ell^-) \\
 &= P(\mathbf{y}_\ell^+ | s_{\ell+1}) \cdot P(s_{\ell+1}, \mathbf{y}_\ell | s_\ell) \cdot P(s_\ell, \mathbf{y}_\ell^-) \quad (11)
 \end{aligned}$$

The second and third lines derive from repeated application of Bayes's Law. The fourth recognizes that only specific dependencies occur, so the others are removed.

As before, we define certain quantities to obtain clearer notation^a.

$$\alpha_\ell(s_\ell) \triangleq P(s_\ell, \mathbf{y}_\ell^-) \quad (12)$$

$$\gamma_\ell(s_\ell, s_{\ell+1}) \triangleq P(s_{\ell+1}, \mathbf{y}_\ell | s_\ell) \quad (13)$$

$$\beta_{\ell+1}(s_{\ell+1}) \triangleq P(\mathbf{y}_\ell^+ | s_\ell) \quad (14)$$

Substituting these into (11) gives

$$P(s_\ell, s_{\ell+1}, \mathbf{y}) = \beta_{\ell+1}(s_{\ell+1}) \cdot \gamma_\ell(s_\ell, s_{\ell+1}) \cdot \alpha_\ell(s_\ell) \quad (15)$$

^aAccording to Heegard and Wicker, the BCJR algorithm is an application of the Baum-Welch algorithm for statistical inference on hidden Markov models. Baum-Welch was reported in L.E. Baum and T. Petrie, “Statistical inference for probabilistic functions of finite state Markov chains,” *Ann. Math. Stat.*, 37:1554-1563, 1966, in which the α, β, γ notation first appeared.

Recursive evolutions:

- Maximizing $L(i_\ell)$ requires enumerating $P(s_\ell, s_{\ell+1}, \mathbf{y})$ over all state pairs in the code trellis. Recursive evolution of α and β with time is needed.
- Let σ_ℓ = the set of all states at time ℓ .
- Then, from the definition:

$$\begin{aligned}
 \alpha_{\ell+1}(s_{\ell+1}) &= P(s_{\ell+1}, \mathbf{y}_{\ell+1}^-) \\
 &= \sum_{s_\ell \in \sigma_\ell} P(s_\ell, s_{\ell+1}, \mathbf{y}_{\ell+1}^-) \\
 &= \sum_{s_\ell \in \sigma_\ell} P(s_{\ell+1}, \mathbf{y}_\ell^- | s_\ell, \mathbf{y}_\ell^-) \cdot P(s_\ell, \mathbf{y}_\ell^-) \\
 &= \sum_{s_\ell \in \sigma_\ell} P(s_{\ell+1}, \mathbf{y}_\ell^- | s_\ell) \cdot P(s_\ell, \mathbf{y}_\ell^-)
 \end{aligned}$$

Or, finally

$$\alpha_{\ell+1}(s_{\ell+1}) = \sum_{s_\ell \in \sigma_\ell} \gamma_\ell(s_\ell, s_{\ell+1}) \cdot \alpha_\ell(s_\ell) \quad (16)$$

(16) is a **forward recursion** of $\alpha_\ell(s_\ell)$ from $\ell \rightarrow \ell + 1$.

Similarly, from (14), a similar analysis shows

$$\beta_\ell(s_\ell) = \sum_{s_{\ell+1} \in \sigma_{\ell+1}} \gamma_\ell(s_\ell, s_{\ell+1}) \beta_{\ell+1}(s_{\ell+1}) \quad (17)$$

and (13.2) is a **backward recursion** of $\beta_{\ell+1}(s_{\ell+1})$ from $\ell + 1$ to ℓ .

(Proof is left as exercise for the reader...)

Initial conditions:

At $\ell = 0$,

$$\alpha_0(s_0) = \begin{cases} 1 & \text{if } s = \mathbf{0} \\ 0 & \text{if } s \neq \mathbf{0} \end{cases}$$

i.e., the sequence begins in the zero state of the trellis.

If h information blocks were transmitted, then at $\ell = K = h + m - 1$,

$$\beta_K(s_\ell) = \begin{cases} 1 & \text{if } s = \mathbf{0} \\ 0 & \text{if } s \neq \mathbf{0} \end{cases}$$

i.e., the path through the trellis must end in the zero state.

The Branch Metric:

$$\begin{aligned}
 \gamma_\ell(s_\ell, s_{\ell+1}) &\triangleq P(s_{\ell+1}, \mathbf{y}_\ell | s_\ell) \\
 &= \frac{P(s_\ell, s_{\ell+1}, \mathbf{y}_\ell)}{P(s_\ell)} \cdot \frac{P(s_\ell, s_{\ell+1})}{P(s_\ell, s_{\ell+1})} \\
 &= \frac{P(s_\ell, s_{\ell+1})}{P(s_\ell)} \cdot \frac{P(s_\ell, s_{\ell+1}, \mathbf{y}_\ell)}{P(s_\ell, s_{\ell+1})} \\
 &= P(s_{\ell+1} | s_\ell) \cdot P(\mathbf{y}_\ell | s_\ell, s_{\ell+1}) \\
 &= P(i_\ell) \cdot P(\mathbf{y}_\ell | \mathbf{x}_\ell)
 \end{aligned} \tag{18}$$

For the AWGN channel,

$$\gamma_\ell(s_\ell, s_{\ell+1}) = P(i_\ell) \left(\sqrt{\frac{E_s}{\pi N_0}} \right)^n e^{-\frac{E_s}{N_0} \|\mathbf{y}_\ell - \mathbf{x}_\ell\|^2} \tag{19}$$

- $\gamma_\ell(s_\ell, s_{\ell+1})$ is a factor of every term of $L(i_\ell)$ (10); so $\left(\sqrt{\frac{E_s}{\pi N_0}} \right)^{nh}$ is a factor of every term in the numerator & denominator of (10);

- therefore, we drop that constant factor and define a **modified branch metric**

$$\gamma_\ell(s_\ell, s_{\ell+1}) = P(i_\ell) e^{-\frac{E_s}{N_0} \|\mathbf{y}_\ell - \mathbf{x}_\ell\|^2} \quad (20)$$

13.3 Log-Domain BCJR Decoder for CCs

Here, we will

1. Derive useful **forward** and **backward** metrics.
2. Derive useful **branch** metrics.
3. Express $L(i_\ell)$ in terms of these metrics.
4. Give the Log-Domain BCHR Decoder.

Following discussion of some schemes for simplifying computations, we will explore use of the BCJR decoder in the Turbo Decoder.

13.3.1 Simplify the branch metric

$$\begin{aligned}
 P(i_\ell = +1) &= \frac{P(i_\ell = +1)}{P(i_\ell = -1) + P(i_\ell = +1)} \\
 &= \frac{P(i_\ell = +1)/P(i_\ell = -1)}{1 + P(i_\ell = +1)/P(i_\ell = -1)}
 \end{aligned}$$

$$\begin{aligned}
 P(i_\ell = -1) &= \frac{P(i_\ell = -1)}{P(i_\ell = -1) + P(i_\ell = +1)} \\
 &= \frac{P(i_\ell = -1)/P(i_\ell = +1)}{1 + P(i_\ell = -1)/P(i_\ell = +1)} \\
 &= \frac{[P(i_\ell = +1)/P(i_\ell = -1)]^{-1}}{1 + [P(i_\ell = +1)/P(i_\ell = -1)]^{-1}}
 \end{aligned}$$

Combine

$$P(i_\ell = \pm 1) = \frac{[P(i_\ell = +1)/P(i_\ell = -1)]^{\pm 1}}{1 + [P(i_\ell = +1)/P(i_\ell = -1)]^{\pm 1}} \quad (21)$$

Substitute definition of $L(i_\ell)$.

$$P(i_\ell = \pm 1) = \frac{e^{\pm L(i_\ell)}}{1 + e^{\pm L(i_\ell)}} \quad (22)$$

But

$$\begin{aligned} P(i_\ell = -1) &= \frac{e^{-L(i_\ell)}}{1 + e^{-L(i_\ell)}} \\ &= \frac{e^{-L(i_\ell)/2} \cdot e^{-L(i_\ell)/2}}{1 + e^{-L(i_\ell)}} \\ &= \frac{e^{-L(i_\ell)/2} \cdot e^{i_\ell L(i_\ell)/2}}{1 + e^{-L(i_\ell)}} \quad (\text{for } i_\ell = -1) \end{aligned} \quad (23)$$

But

$$P(i_\ell = +1) + \frac{e^{+L(i_\ell)}}{1 + e^{+L(i_\ell)}} = \frac{1}{1 + e^{-L(i_\ell)}}$$

And substituting $i_\ell = +1$ into (23) gives

$$\frac{e^{-L(i_\ell)/2} \cdot e^{i_\ell L(i_\ell)/2}}{1 + e^{-L(i_\ell)}} = \frac{1}{1 + e^{-L(i_\ell)}}.$$

Let

$$A_\ell = \frac{e^{-L(i_\ell)/2}}{1 + e^{-L(i_\ell)}}.$$

Then

$$P(i_\ell = \pm 1) = A_\ell \cdot e^{i_\ell \cdot L(i_\ell)/2}. \quad (24)$$

A_ℓ is independent of i_ℓ , so (24) can be used for $P(i_\ell)$ in the modified branch metric (20). Specifically,

1. For $\ell = 0, 1, \dots, h - 1$ (where h is the number of information blocks [bits] transmitted):

$$\begin{aligned}\gamma_\ell(s_\ell, s_{\ell+1}) &= A_\ell \cdot e^{i_\ell L(i_\ell)/2} e^{-\frac{E_s}{N_0} \|\mathbf{y}_\ell - \mathbf{x}_\ell\|^2} \\ &= A_\ell B_\ell \cdot e^{i_\ell L(i_\ell)/2} e^{-\frac{2E_s}{N_0} (\mathbf{y}_\ell \cdot \mathbf{x}_\ell)}\end{aligned}\quad (25)$$

2. For the termination bits, $\ell = h, h + 1, \dots, n + m - 1 = K - 1$, $P(i_\ell) = 1$ and $L(i_\ell) = \pm\infty$ as $i_\ell = \pm 1$:

$$\gamma_\ell(s_\ell, s_{\ell+1}) = B_\ell \cdot e^{-\frac{2E_s}{N_0} (\mathbf{y}_\ell \cdot \mathbf{x}_\ell)}\quad (26)$$

Further simplification:

As before, $\prod_{\ell=0}^{h-1} A_\ell$ and $\prod_{\ell=0}^{K-1} B_\ell$ can be divided out of the $L(\cdot)$ function (8):

$$\begin{aligned}\lambda_\ell(s_\ell, s_{\ell+1}) &= e^{i_\ell L(i_\ell)/2} \cdot e^{(2E_s/N_0)(\mathbf{y}_\ell \cdot \mathbf{x}_\ell)} \quad \ell = 0, \dots, h - 1 \\ &= e^{(2E_s/N_0)(\mathbf{y}_\ell \cdot \mathbf{x}_\ell)}, \quad \ell = h, \dots, K - 1\end{aligned}$$

Note: Often, you will see $L_c \triangleq 4E_s/N_0$, the “channel reliability factor.”

Simplification in the log domain

$$\begin{aligned} \gamma_\ell^*(s_\ell, s_{\ell+1}) &\triangleq \ln \gamma_\ell(s_\ell, s_{\ell+1}) \\ &= \frac{i_\ell L(i_\ell)}{2} + \frac{2E_s}{N_0} (\mathbf{y}_\ell \cdot \mathbf{x}_\ell), \quad \ell = 0, 1, \dots, h-1 \end{aligned} \quad (27)$$

$$= \frac{2E_s}{N_0} (\mathbf{y}_\ell \cdot \mathbf{x}_\ell), \quad \ell = h, \dots, K-1 \quad (28)$$

To simplify sums of exponentials, define

$$\begin{aligned} \max^*(x, y) &\triangleq \ln(e^x + e^y) \\ &= \max(x, y) + \ln(1 + e^{-|x-y|}), \end{aligned}$$

so as to replace logs of sums of exponentials with simple max and log functions.

$$\begin{aligned}
\alpha_{\ell+1}^*(s_{\ell+1}) &\triangleq \ln \alpha_{\ell+1}(s_{\ell+1}) \\
&= \ln \sum_{s_\ell \in \sigma_\ell} \gamma_\ell(s_\ell, s_{\ell+1}) \alpha_\ell(s_\ell) \\
&= \ln \sum_{s_\ell \in \sigma_\ell} e^{\gamma_\ell^*(s_\ell, s_{\ell+1})} \alpha_\ell^*(s_\ell) \\
&= \max^*_{s_\ell \in \sigma_\ell} [\gamma_\ell^*(s_\ell, s_{\ell+1}) + \alpha_\ell^*(s_\ell)], \quad \ell = 0, \dots, K-1 \quad (29)
\end{aligned}$$

$$\alpha_0^*(s_{\ell+1}) \triangleq \ln \alpha_0(s_{\ell+1}) = \begin{cases} 0, & s_{\ell+1} = \mathbf{0} \\ -\infty, & s_{\ell+1} \neq \mathbf{0} \end{cases}$$

Similarly,

$$\begin{aligned}
\beta_\ell^*(s_\ell) &\triangleq \ln \beta_\ell(s_\ell) \\
&= \ln \sum_{s_{\ell+1} \in \sigma_{\ell+1}} \gamma_\ell(s_\ell, s_{\ell+1}) \beta_{\ell+1}(s_{\ell+1}) \\
&= \ln \sum_{s_{\ell+1} \in \sigma_{\ell+1}} e^{\gamma_\ell^*(s_\ell, s_{\ell+1})} \beta_{\ell+1}^*(s_{\ell+1}) \\
&= \max_{s_{\ell+1} \in \sigma_{\ell+1}}^* [\gamma_\ell^*(s_\ell, s_{\ell+1}) + \beta_{\ell+1}^*(s_{\ell+1})], \\
&\quad \ell = K - 1, K - 2, \dots, 0
\end{aligned} \tag{30}$$

$$\beta_K^*(s_{\ell+1}) \triangleq \ln \beta_K(s_{\ell+1}) = \begin{cases} 0, & s_{\ell+1} = \mathbf{0} \\ -\infty, & s_{\ell+1} \neq \mathbf{0} \end{cases} \tag{31}$$

So, the joint probability of the states and the received vector can be written in terms of the foregoing as

$$P(s_\ell, s_{\ell+1}, \mathbf{y}) = e^{\beta_{\ell+1}^*(s_{\ell+1}) + \gamma_\ell^*(s_\ell, s_{\ell+1}) + \alpha_\ell^*(s_\ell)} \quad (32)$$

From (32), the expression for $L(i_\ell)$ can be written in the obvious manner; then the \max^* operator can be used to get:

$$\begin{aligned} L(i_\ell) = & \max_{s_\ell, s_{\ell+1} \in \Sigma_\ell^+}^* [\beta_{\ell+1}^*(s_{\ell+1}) + \gamma_\ell^*(s_\ell, s_{\ell+1}) + \alpha_\ell^*(s_\ell)] \\ & - \max_{s_\ell, s_{\ell+1} \in \Sigma_\ell^-}^* [\beta_{\ell+1}^*(s_{\ell+1}) + \gamma_\ell^*(s_\ell, s_{\ell+1}) + \alpha_\ell^*(s_\ell)] \quad (33) \end{aligned}$$

Log-domain BCJR Algorithm (AKA log-MAP algorithm)

Working in the log domain and using a $\max(\cdot)$ function is a less computationally intensive way to evaluate (33) than more direct methods and offers greater numerical stability.

1. Initialize forward and backward metrics $\alpha_0^*(s_{\ell+1})$ and $\beta_K^*(s_{\ell+1})$ using (30) and (31), respectively.
2. Compute branch metrics $\gamma_\ell^*(s_\ell, s_{\ell+1}), \ell = 0, 1, \dots, K - 1$ using (27) and (28).
3. Compute forward metrics using (29).
4. Compute backward metrics using (30).
5. Compute the MAP values $L(i_\ell), \ell = 0, 1, \dots, h - 1$.

If desired, compute hard decisions $\hat{i}_\ell \in \{\pm 1\}$ using the signs of the $L(i_\ell)$.

Comparisons

- Steps 3, 4, and 5 **each** require about the same amount of computation as the Viterbi algorithm (WHY?). Hence, BCJR is about 3x more compute intensive than VA.
- BCJR requires knowledge of the channel SNR; Viterbi does not, merely computing branch correlations.
- The BCJR is preferable for iterative (i.e., turbo) decoding.