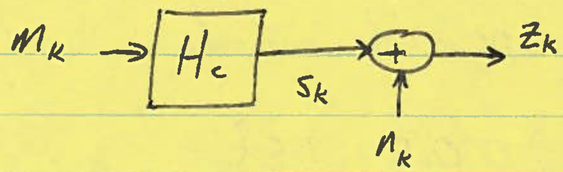
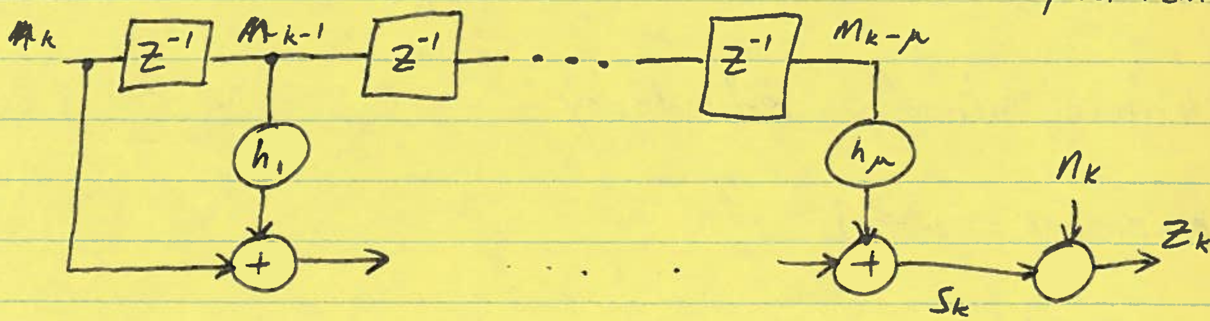
L13 Sequence Detection

13.1 Channel Response

- after channel + electronics



- generally a system with memory .. μ (# of previous symbols ~~repem~~ remembered)



- the output is a convolution in discrete time

$$s_k = \sum_{l=0}^{\mu} h_l m_{k-l} = \sum_{l=0}^{\mu} h_{k-l} m_l$$

- effectively the output depends on...

$$s_k = f(m_k, \bar{\psi}_k)$$

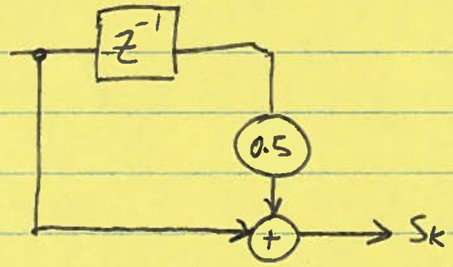
current input — current state of channel

$\bar{\psi}_k = [m_{k-1}, m_{k-2}, \dots m_{k-\mu}]$

- or entirely in terms of current and upcoming states

$$s_k = g(\bar{\psi}_k, \bar{\psi}_{k+1})$$ ← o/p is fn. of a state transition

e.g.



- for alphabet $\{0,1\}$ of size $M=2$
- possible outputs

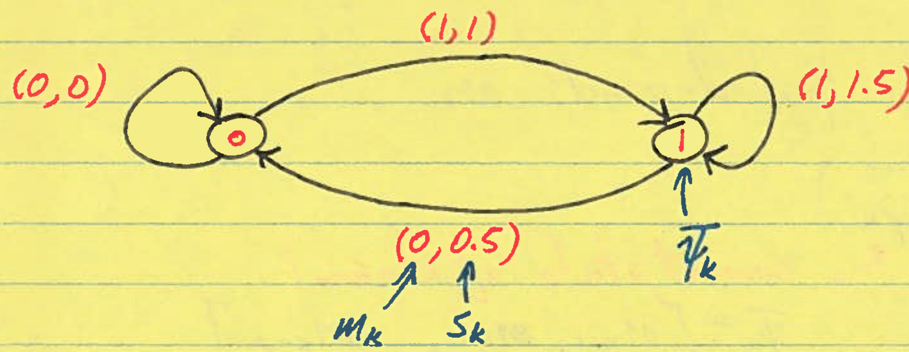
m_k ψ_k

$$s_k = \{s_k^{00}, s_k^{01}, s_k^{10}, s_k^{11}\} = \{0, 0.5, 1, 1.5\}$$

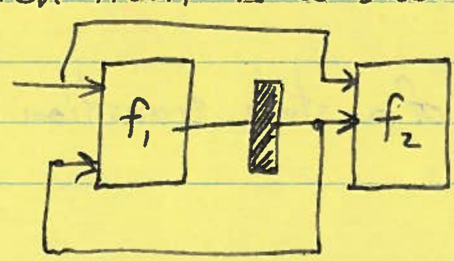
- 1-bit of info comes in + one of 4 levels comes out
- channel introduces redundancy
- memory $= \mu = 1$

13.2 FSMs

- A generic representation of our channel is in the form of a state transition diagram



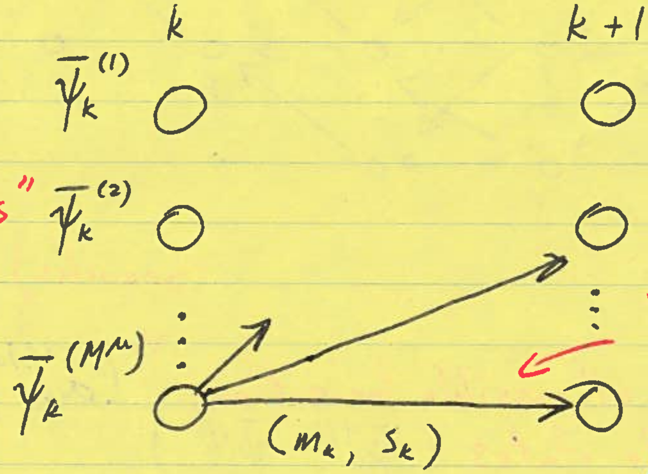
- which itself is a description of a FSM



13.3 Trellis Diagram

- Alternative to traditional state transition diagram
- Shows all possible progressions of states over time

represent states with circles: "nodes"
M^μ nodes per k

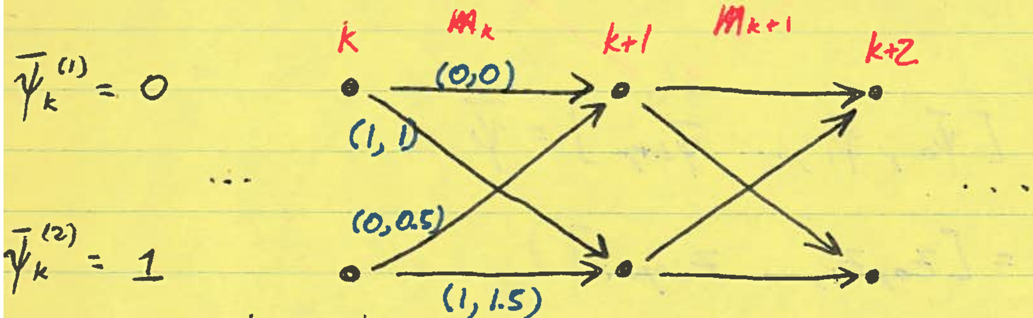


"branch": arcs between states correspond to a particular state transition at a particular time
M branches per state

- in general

$$\bar{\psi}_k^{(i)} \xrightarrow{\left(m_k^{(i,j)},\ s_k^{(i,j)}\right)} \psi_{k+1}^{(j)}$$

e.g. our simple channel in trellis form

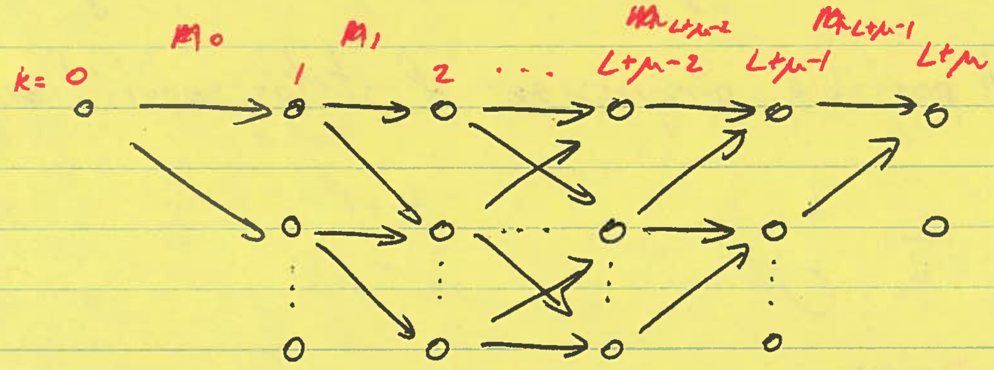


ψ_k ψ_{k+1}

$$m^{(0,0)} = 0, \quad m^{(0,1)} = 1, \quad m^{(1,0)} = 0, \quad m^{(1,1)} = 1$$

$$s^{(0,0)} = 0 \quad s^{(0,1)} = 1, \quad s^{(1,0)} = 0.5, \quad s^{(1,1)} = 1.5$$

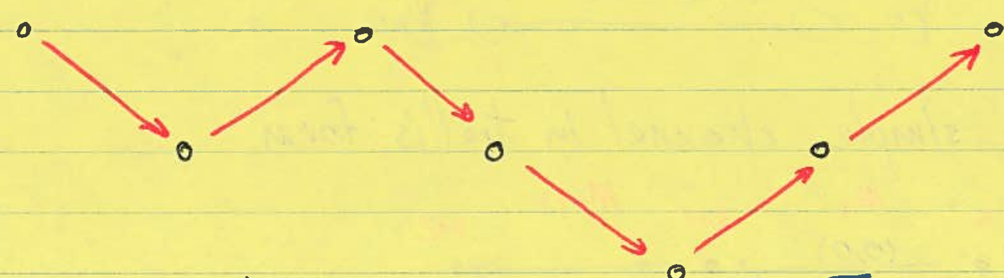
- bounds are often imposed by assuming that the sequence starts at some known state and ends at some known state



13.4 Path Detection

- The trellis shows all possible transitions (branches) $(a_k^{(i,j)}, s_k^{(i,j)})$ between all possible states $(\bar{\psi}_k^{(i)}, \bar{\psi}_{k+1}^{(j)})$

- ~~Only one~~ combination of branches across the trellis constitutes the actual path taken by the signal $\bar{m} = [m_0, m_1, \ldots, m_{L+\mu}$



through the states $[\bar{\psi}_0, \bar{\psi}_1, \ldots, \bar{\psi}_{L+\mu}] = \bar{\psi}$

(and received as $\bar{z} = [z_0, z_1, \ldots z_{L+\mu-1}]$)

- How do we find that path? (i.e. the route through the trellis states defined by the state sequence $\bar{\psi}$)

- "easy" use...

... MAP

$$\hat{\bar{\psi}} = \arg\max_{\bar{\psi}} p(\bar{\psi} | \bar{z})$$

$$p(\bar{\psi} | \bar{z}) = \underbrace{p(\bar{z} | \bar{\psi})}\,\underbrace{p(\bar{\psi})}$$

$$p(\bar{z} | \bar{\psi}) = \prod_{k=0}^{L+\mu-1} p_z(z_k | \bar{\psi})$$

probability of sequence is product of probabilities of constituent transistions

further, z_k depends only on adjacent (in time) states so...

$$p(\bar{z} | \bar{\psi}) = \prod_{k=0}^{L+\mu-1} p_z(z_k | \bar{\psi}_k, \bar{\psi}_{k+1})$$

$$p(\bar{\psi}) = p(\psi_0) \prod_{k=0}^{L+\mu-1} p(\bar{\psi}_{k+1} | \bar{\psi}_k)$$

product of probabilities of corresponding state transitions (and initial state)

$$\hat{\bar{\psi}} = \arg\max_{\bar{\psi}} \prod_{k=0}^{L+\mu-1} \left[p_z(z_k | \bar{\psi}_k, \bar{\psi}_{k+1}) \, p(\bar{\psi}_{k+1} | \bar{\psi}_k) \right]$$

for $\bar{\psi}_k^{(i)} = i$ and $\bar{\psi}_{k+1}^{(j)} = j$, recall $s_k = g(\bar{\psi}_k, \bar{\psi}_{k+1})$,

$$= \arg\max_{\bar{\psi}} \left\{ \prod_{k=0}^{L+\mu-1} \underbrace{p_z(z_k | s^{(i,j)}) \cdot p_{m_k}(m^{(i,j)})}_{\text{branch metrics}} \right\}$$

$p(\bar{\psi}_{k+1} | \bar{\psi}_k)$ depends on input m_k

- the path value consists of $L+\mu$ branch metrics

$$\beta_k(i,j) = p_Z\left(z_k \mid s^{(i,j)}\right) \cdot p_{m_k}\left(m^{(i,j)}\right)$$

branch metric @ time k from states i to j

- for Gaussian noise

$$\beta_k(i,j) = p_N\left(z_k - s^{(i,j)}\right) \cdot p_{m_k}\left(m^{(i,j)}\right)$$

$$= \frac{1}{\sqrt{2\pi\sigma_0^2}}\, e^{-\frac{|z_k - s^{(i,j)}|^2}{2\sigma_0^2}} \cdot p_{m_k}\left(m^{(i,j)}\right)$$

$$-\log\left(\beta_k(i,j)\right) = \frac{|z_k - s^{(i,j)}|^2}{2\sigma_0^2} + \frac{1}{2}\log\left(2\pi\sigma_0^2\right) - \log\left[p_{m_k}\left(m^{(i,j)}\right)\right]$$

$$B_k(i,j)\Big|_{MAP} = |z_k - s^{(i,j)}|^2 + \sigma_0^2\log\left(2\pi\sigma_0^2\right) - 2\sigma_0^2\log\left[p_{m_k}\left(m^{(i,j)}\right)\right]$$

- for ML all $m^{(i,j)}$ are equally likely hence $p_{m_k}(m^{(i,j)})$ same for all branches

- σ_0^2 same for all branches as well and hence these two components are just offsets which the maximum likelihood sequence detector (MLSD) can ignore

$$B_k(i,j)\Big|_{ML} = |z_k - s^{(i,j)}|^2$$

- because of our manipulations our path metric becomes

$$P\Big|_{ML} = \sum_{k=0}^{L+\mu-1} |z_k - s^{(i,j)}|^2 \quad : \text{sum of squares}$$

our "cost function"

and...

... $\hat{\psi} = \arg\min_{\bar{\psi}} P|_{ML}$ ← that is our ML path selection criterion is a least squares criterion

13.5 Viterbi Algorithm

Finding the sequence that minimizes...

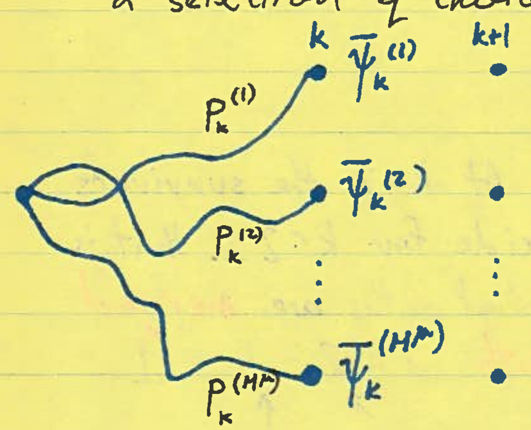
$$P|_{ML} = \sum_{k=0}^{L+\mu-1} |z_k - s^{(i,j)}|^2 \quad \text{... is a tall order}$$

- we have M^{μ} states
- M branches leaving & entering each state
- L time intervals

therefore about $(M^{\mu+1})^L$ possible sequences

- luckily this is amenable to minimization via dynamic programming

  ↑ i.e. finding the min path metric
- a means of iteratively computing a sequence of steps from a selection of choices



1. identify optimal paths to states at time k, $P_k^{(i)}$ (partial paths) with their partial path metrics
2. calculate all branch metrics from k to k+1 : $B_k(i,j)$ [i.e. M partial paths at each state $\bar{\psi}_{k+1}^{(j)}$...

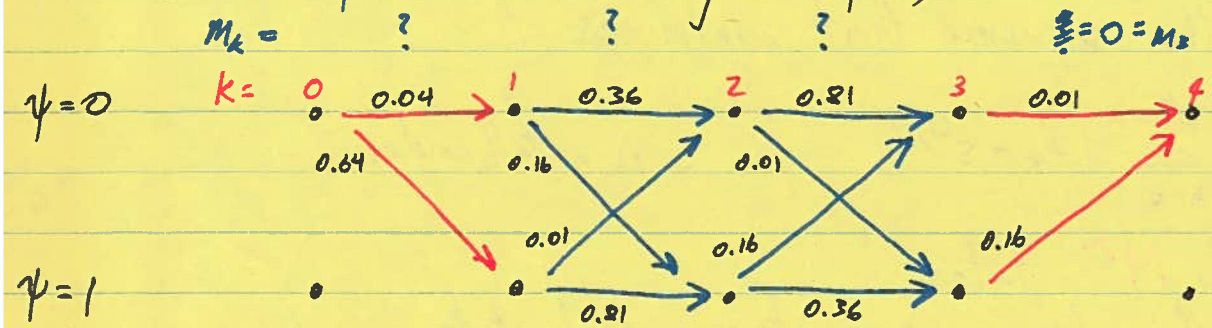
$$P'^{(j)}_{k+1} = \{\underbrace{P_k^{(i)} + B_k(i,j)}_{\text{set of } M \text{ partial paths into ea. state } \psi_{k+1}^{(j)}}\}$$

3. choose survivor path at each stage

$$P_{k+1}^{(j)} = \min\{P_k^{(i)} + B_k(i,j)\}$$

- in the case where the trellis starts and ends in a known state (i.e. m_{-1} is known as are m_L to $m_{L+\mu}$) the remaining path (i.e. the survivor sequence at time $L+\mu$) is the optimal path

- for example (our running example)

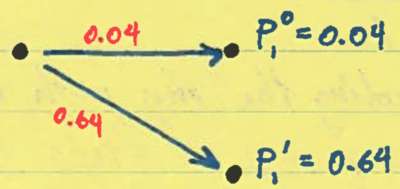


branch labels are branch metrics

$z_k =$ 0.2 0.6 0.9 0.1 ← what you actually received

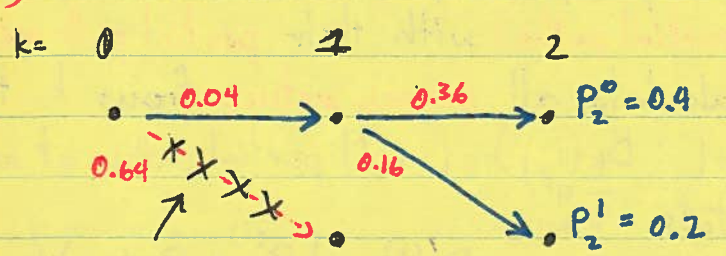
- try to find the most likely path...

1) survivors at $k=1$



only one path into ea. state at $k+1$, no decisions to make

2) survivors at $k=2$

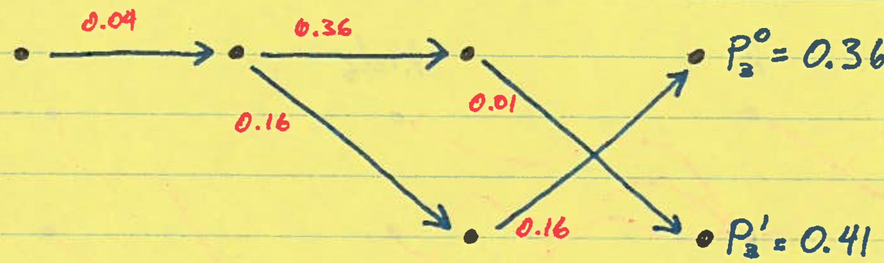


don't even bother expanding this path from $k=1$ as the $B_1(i,j)$'s from the other state are < 0.64

NOTE: At $k=2$ the survivors all coincide for $k<2$... that is the partial paths are merged at depth $d = 2 - 1 = 1$

$k=2$ $k=1$

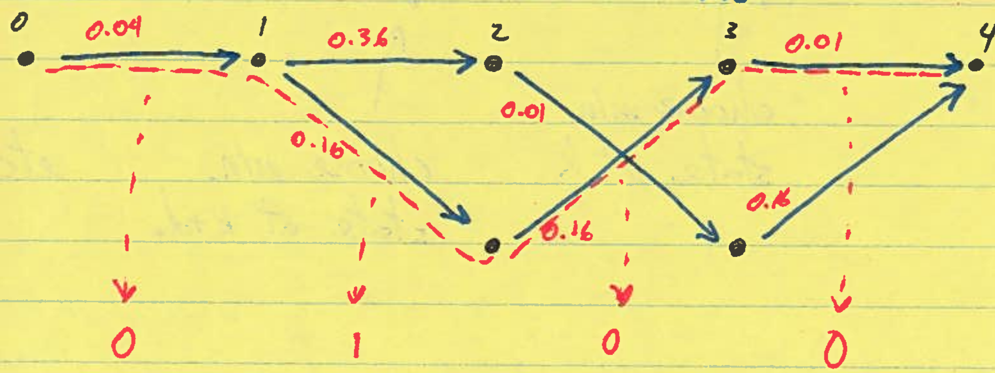
3) survivors at $k = 3$



$d = 2$
(merge depth)

4) survivors at $k = 4$

• know that $m_3 = 0$



$d = 3$

parting thoughts

- having to wait for a whole sequence to finish can be quite restricting
- path merges are nice because they allow us to start with a new sequence, but they are not guaranteed to happen so we can't expect to wait around for them
- in practice you can always choose to employ some truncation depth d_t

- at k find cheapest path and use that as your origin... repeat every d_t time steps