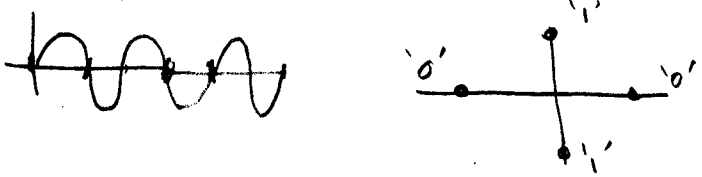
6.5 Detecting Signals With Memory – the Viterbi Algorithm

P+S 7.4.2

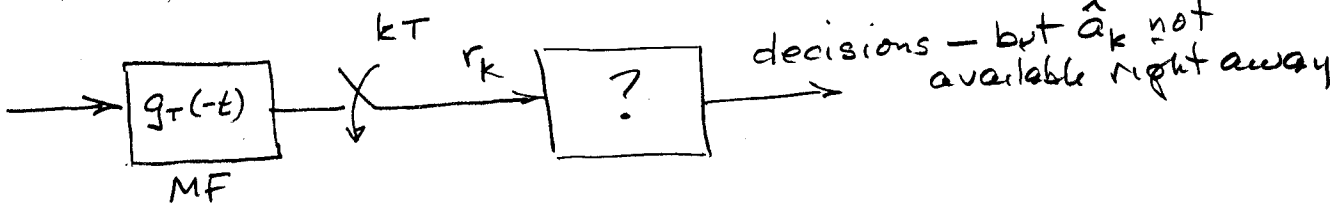
- When we detect signals with memory, we have two choices:
  - Ignore the memory and take a hit on BER vs SNR. We did this for AMI & bipolar in previous section. Similarly, treat this $h=\frac{1}{2}$ 2FSK as biorthogonal with duplicates



  - Account for the memory. More complex, but significant performance improvement. That's what we learn in the present section, using the celebrated Viterbi Algorithm.

- When we account for memory, we use the facts that some sequences cannot occur, and the remainder have, in principle, a new pdf. Most of the applications have all allowable sequences equiprobable. Since we are now detecting entire sequences, the approach is MLSE (max likelihood sequence estimation).

- The detector in white noise is still constructed with a matched filter or correlator front end:



- In each symbol interval, we still have conditional probabilities $p(r|s_1)$, $p(r|s_2)$. For example, binary antipodal:

$$s_1 = \sqrt{E_b}, \quad s_2 = -\sqrt{E_b}$$

$$p(r|s_1) = \frac{1}{\sqrt{2\pi}\sigma_n} e^{-\frac{1}{2}\left(\frac{r-\sqrt{E_b}}{\sigma_n}\right)^2} \qquad p(r|s_2) = \frac{1}{\sqrt{2\pi}\sigma_n} e^{-\frac{1}{2}\left(\frac{r+\sqrt{E_b}}{\sigma_n}\right)^2}$$

and for AMI

$$s_1 = \sqrt{2E_b} \qquad s_2 = 0 \qquad s_3 = -\sqrt{2E_b}$$

$$p(r|s_1) = \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(-\frac{1}{2}\left(\frac{r-\sqrt{2E_b}}{\sigma_n}\right)^2\right) \qquad p(r|s_2) = \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(-\frac{1}{2}\left(\frac{r}{\sigma_n}\right)^2\right)$$

$$p(r|s_3) = \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left(-\frac{1}{2}\left(\frac{r+\sqrt{2E_b}}{\sigma_n}\right)^2\right)$$

- Sequence estimation. Define $\underline{r}_K = (r_1, r_2, \cdots, r_K)$ and $\underline{\sigma}_K = (\sigma_1, \sigma_2, \cdots \sigma_K)$ where σ_i is the state at time i. Since the state sequence determines the data sequence, and the state sequences are equiprobable, we do this:

$$\underset{\underline{\sigma}_K}{\arg\max}\ p(\underline{r}_K|\underline{\sigma}_K) \quad \text{for length-}K\text{ sequences.}$$

This is ML. Suppose there are 2 states. It looks as though we have to run through all 2^K sequences $\underline{\sigma}$, compute $p(\underline{r}_K|\underline{\sigma}_K)$ for each one and find the largest. Is this realistic?

- A recursive formulation is our first step out of the swamp of exponential growth.

$$p(\underline{r}_K \mid \underline{\sigma}_K) = p(r_K, \underline{r}_{K-1} \mid \underline{\sigma}_K) = p(r_K \mid \underline{r}_{K-1}, \underline{\sigma}_K)\, p(\underline{r}_{K-1} \mid \underline{\sigma}_K) \quad \text{why?}$$

$$= p(r_K \mid \underline{\sigma}_K)\, p(\underline{r}_{K-1} \mid \underline{\sigma}_{K-1}) \quad \text{why?}$$

$$= p(r_K \mid \sigma_K, \sigma_{K-1})\, p(\underline{r}_{K-1} \mid \underline{\sigma}_{K-1}) \quad \text{why?}$$

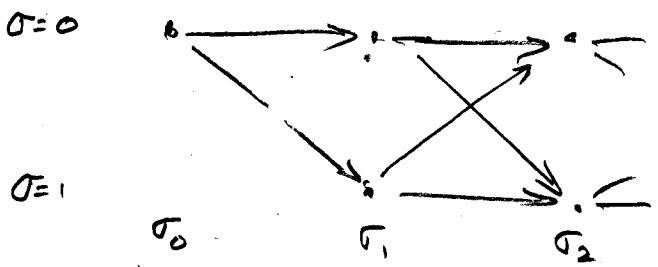
$$= p(r_K \mid \sigma_K, \sigma_{K-1})\, p(r_{K-1} \mid \sigma_{K-1}, \sigma_{K-2}) \cdots p(r_2 \mid \sigma_2, \sigma_1)\, p(r_1 \mid \sigma_1, \sigma_0)$$

and the logarithm converts it to an additive form

$$\ln(p(\underline{r}_K \mid \underline{\sigma}_K) = \ln(p(r_K \mid \sigma_K, \sigma_{K-1})) + \ln(p(r_{K-1} \mid \sigma_{K-1}, \sigma_{K-2})) + \cdots + \ln(p(r_1 \mid \sigma_1, \sigma_0))$$

from previous page $\ln(p(r_i \mid \sigma_i, \sigma_{i-1}) \sim -(r - s(\sigma_i, \sigma_{i-1}))^2$ — a function that gives s from the state pair

so $\underset{\underline{\sigma}_K}{\operatorname{argmax}} \ln(p(\underline{r}_K \mid \underline{\sigma}_K)) = \underset{\underline{\sigma}_K}{\operatorname{argmin}} \sum_{i=1}^{K} (r_i - s(\sigma_i, \sigma_{i-1}))^2$



We minimize the Euclidean metric.

branch metric

$$\mu(r_i, \sigma_{now}, \sigma_{then}) = (r - s(\sigma_{now}, \sigma_{then}))^2$$

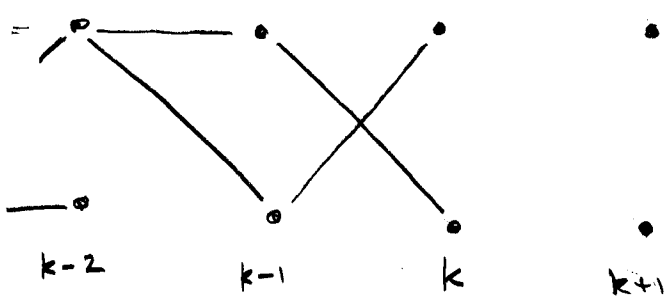
cumulative metric

$$M(\underline{r}) = \sum_{i=1}^{K} \mu(r_i, \sigma_i, \sigma_{i-1})$$

Not much progress so far.

- The "dynamic programming" recursion is the next step. Suppose we know at time k the best paths terminating in each state 0, 1 (denote $\underline{\tilde{\sigma}}_k(0)$, $\underline{\tilde{\sigma}}_k(1)$) and the corresponding metrics $M(\tilde{\sigma}_k(0))$, $M(\tilde{\sigma}_k(1))$.

$\underline{\tilde{\sigma}}_k(0)$, $\underline{\tilde{\sigma}}_k(1)$ are the "survivor" paths at time k

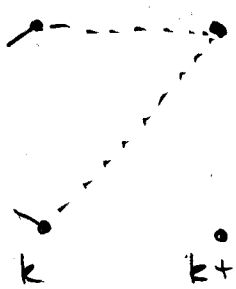


To extend to time $k+1$, note that the best path terminating in $\sigma_{k+1} = 0$ has either $\underline{\tilde{\sigma}}_k(0)$ or $\underline{\tilde{\sigma}}_k(1)$ as a prefix. Same with best path terminating in $\sigma_{k+1} = 1$. That is,

$$\underline{\tilde{\sigma}}_{k+1}(0) = \left[\underline{\tilde{\sigma}}_k(0), 0\right] \text{ or } \left[\underline{\tilde{\sigma}}_k(1), 0\right]$$

$$\underline{\tilde{\sigma}}_{k+1}(1) = \left[\underline{\tilde{\sigma}}_k(0), 1\right] \text{ or } \left[\underline{\tilde{\sigma}}_k(1), 1\right]$$

For state 0:



Choose prefix that minimizes accumulated metric

$$M(\underline{\tilde{\sigma}}_k(0)) + \mu(r_{k+1}, 0, 0)$$

or

$$M(\underline{\tilde{\sigma}}_k(1)) + \mu(r_{k+1}, 1, 0)$$

The smaller value becomes $M(\tilde{\sigma}_{k+1}(0))$.

For state 1:

Choose prefix that minimizes accumulated metric

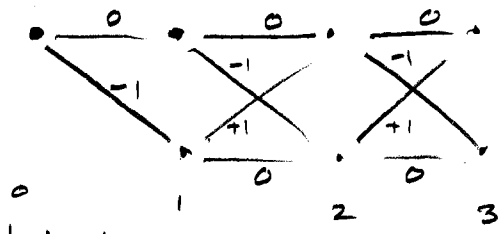
$$M(\tilde{\sigma}_k(0)) + \mu(r_{k+1}, 0, 1)$$

or

$$M(\tilde{\sigma}_k(0)) + \mu(r_{k+1}, 1, 1)$$

The smaller value becomes $M(\tilde{\sigma}_{k+1}(0))$

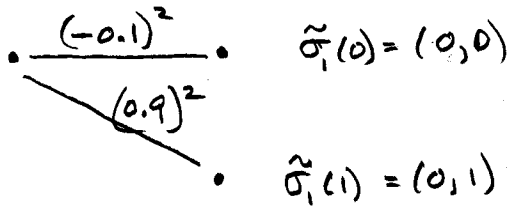
Example AMI, initialized in state 0, $M(\sigma_0=0) = 0$
normalized to amplitude 1, so $E_b = \frac{1}{2}$



We receive

$r_1 = -0.1$, $r_2 = -0.9$, $r_3 = 1.1$

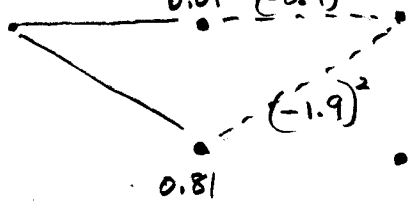
- time k=1



$\tilde{\sigma}_1(0) = (0,0)$

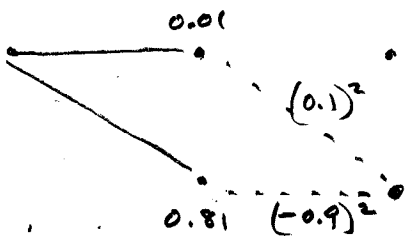
$\tilde{\sigma}_1(1) = (0,1)$

- time k=2



smaller of 0.82 and 4.42 makes 0,0 the better prefix.

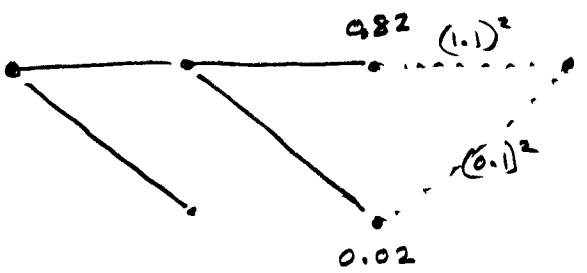
$\tilde{\sigma}_2(0) = (0,0,0)$ $M(\tilde{\sigma}_2(0)) = 0.82$



smaller of 0.02 and 1.62 makes 0,0 the better prefix

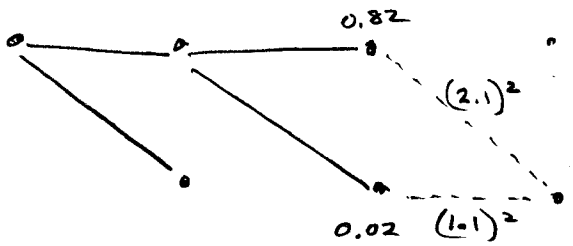
$\tilde{\sigma}_2(1) = (0,0,1)$ $M(\tilde{\sigma}_2(1)) = 0.02$

- time $k=3$



smaller of 2.03 and 0.03 makes 0,0,1 the better prefix

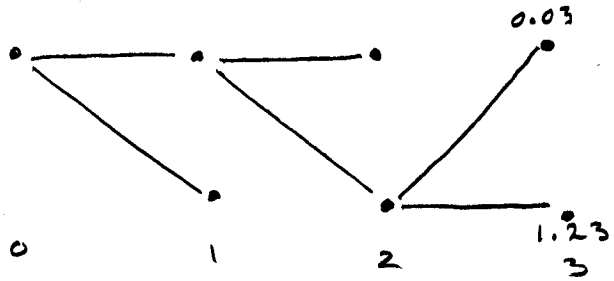
$\tilde{\sigma}_3(0) = (0,0,1,0)$ $M(\tilde{\sigma}_3(0)) = 0.03$



smaller of 5.23 and 1.23 makes 0,0,1 the better prefix

$\tilde{\sigma}_3(1) = (0,0,1,1)$ $M(\tilde{\sigma}_3(1)) = 1.23$

All possible future paths have 0,0,1 as a prefix. Release them as decisions. Termed a "merge" of paths



This particular trellis and constellation can be simplified further, but this example should be enough to establish:

- the method
- the fact that computation is proportional to K not 2^K (!)