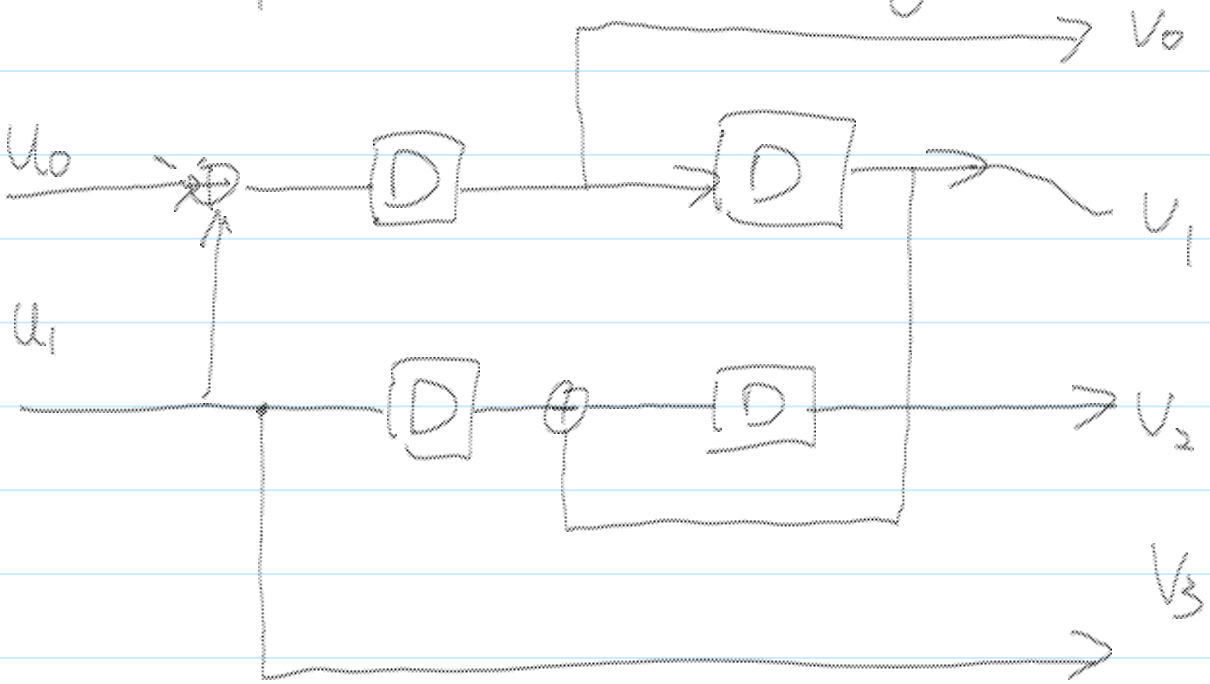
L24

Monday, November 14, 2022 1:39 PM

* Binary convolutional codes. by Elias 1955

A special class of binary linear codes



Features of convolutional codes

① Serial input (single bit or multiple bits.

$u_0\ u_1$ | $u_2\ u_3$ | $u_4\ u_5$ . . .

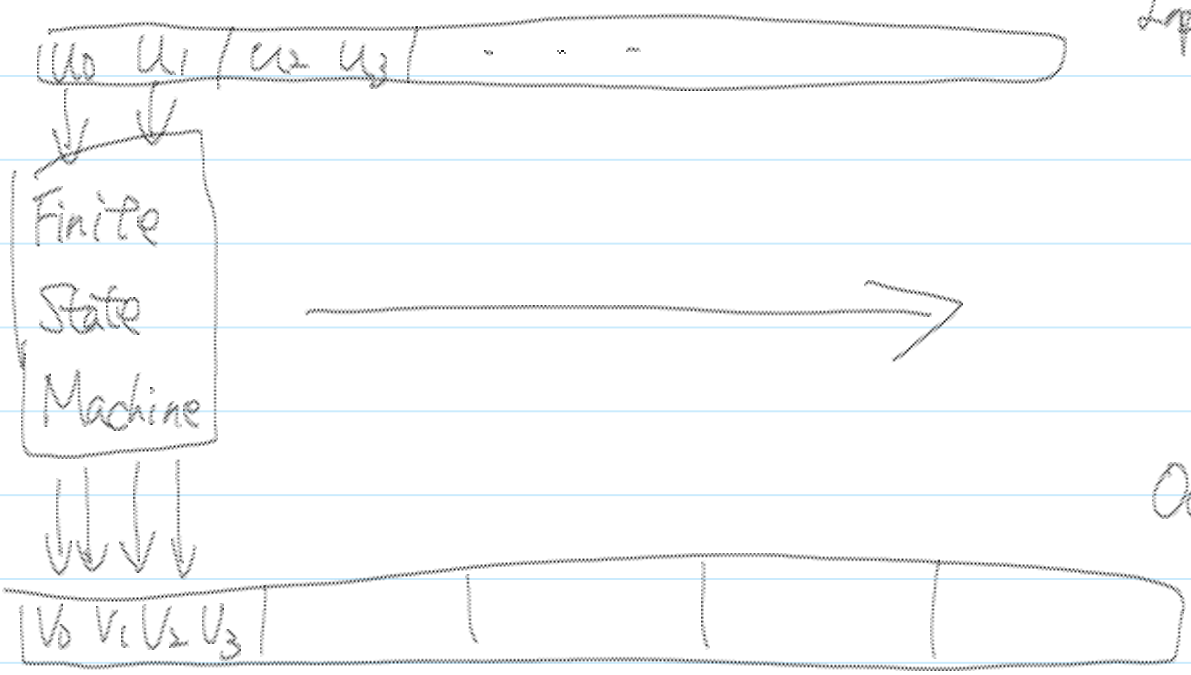
② Serial output

$v_0\ v_1\ v_2\ v_3$ | $v_4\ v_5\ v_6\ v_7$ . . .

③ Arbitrary connection among a finite # of delay units & binary ⊕ operations.

④ All memory or delay units are initialized to zero.

A more general form, sometimes called the trellis code is based on the finite state machine



Properties of the convolutional code

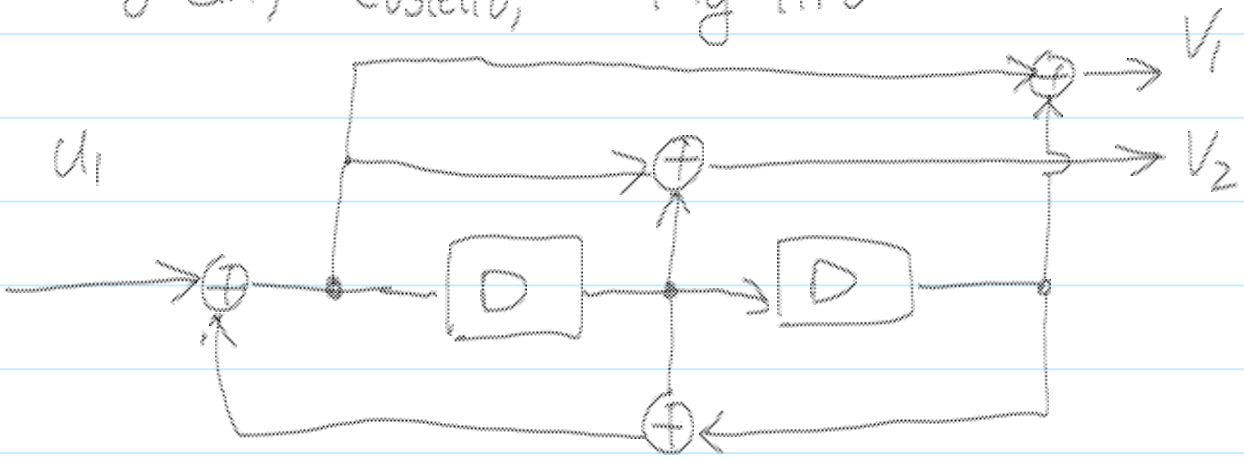
Properties of the convolutional code

① Still a binary linear code. (Since the delay, ⊕ used for the FSM are linear.)

Since the input/output can be viewed as a discrete-time linear time-invariant system

$\Rightarrow P_X$ is uniform.

A running example in Lin, Costello, Fig 11.6



② The code rate is $\frac{\#\text{ of } u \text{ bits}}{\#\text{ of } v \text{ bits}}$

$= \frac{1}{2}$ in our example

③ Encoding complexity: $O(n(\text{Complexity of FSM}))$

★ ④ Complexity of "optimal decoding"

$O(|\#\text{ states}| \times n)$

We will discuss it later.

③+④ ⇒ Scalable for large n, Long codewords have better performance.

⑤ Early pseudo-random number generators are based on shift registers with feedback. (PRNG)

⇒ the $\vec{X}$ codeword distribution looks random

Recall random codes achieve the capacity. We expect good performance of the convolutional codes.

Unfortunately, the seq generated by shift registers is not random enough.

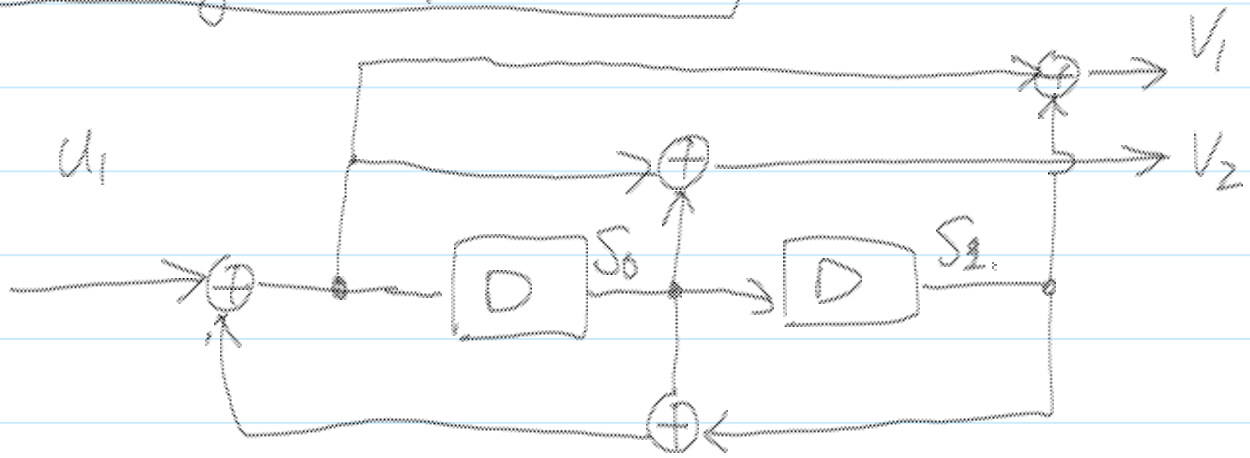
⇒ ☹ Not the best PRNG.

② The performance is still away from capacity.

* Optimal decoding (the Viterbi algorithm)

* A detour to the trellis representation

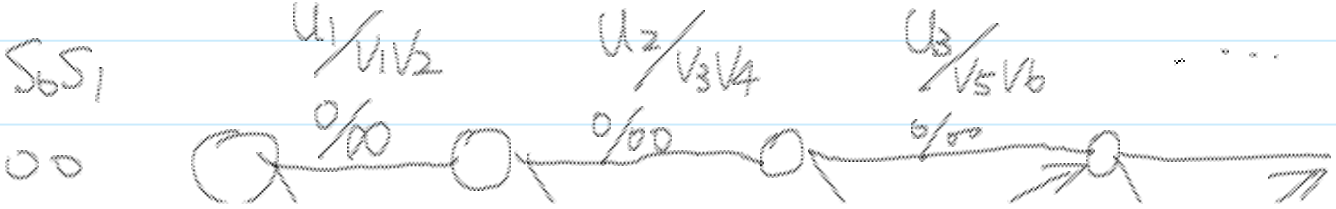
Shift register representation

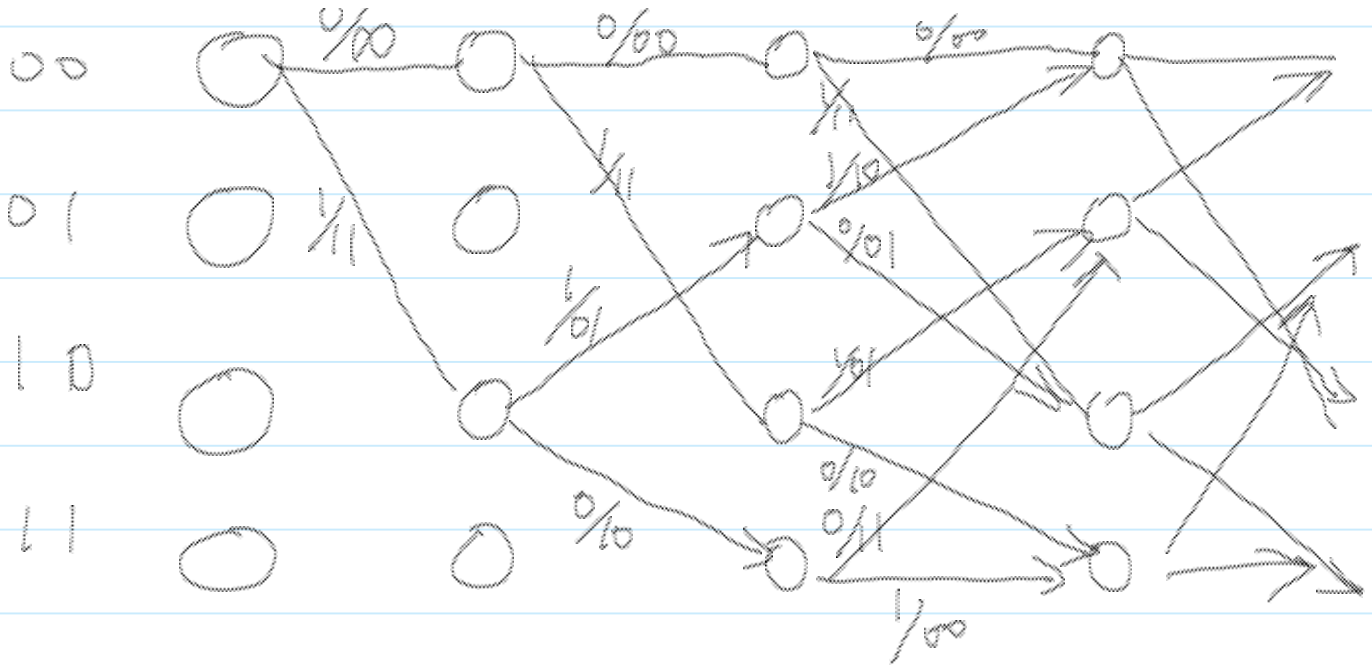


# of possible "states" = $2^{\#\text{ delay units}}$

The state is thus described by $S_0 S_1$, the bit values of the delay units.

Trellis representation





$u_1\ u_2\ u_3 \ldots$: the input message bits that "steer" the path

$V_1V_2\ V_3V_4\ V_5V_6$ the positions of the output bit string that correspond to the selected segment of the path.

* The information bits steer the path along the trellis structure

* Each path corresponds to a codeword

Codeword

* code rate: $\frac{\#\text{ of } u \text{ bits}}{\#\text{ of } v \text{ bits}}$

Our example is a rate $\frac{1}{2}$ convolutional code

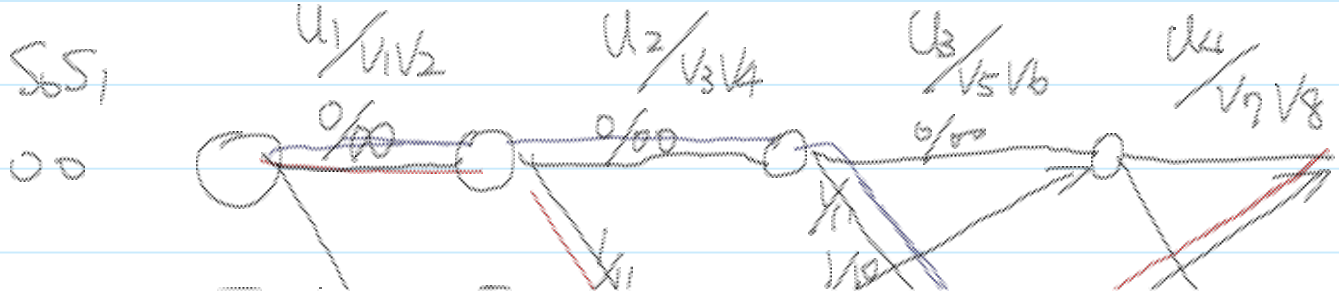
Optimal MAP decoder for the convolutional code

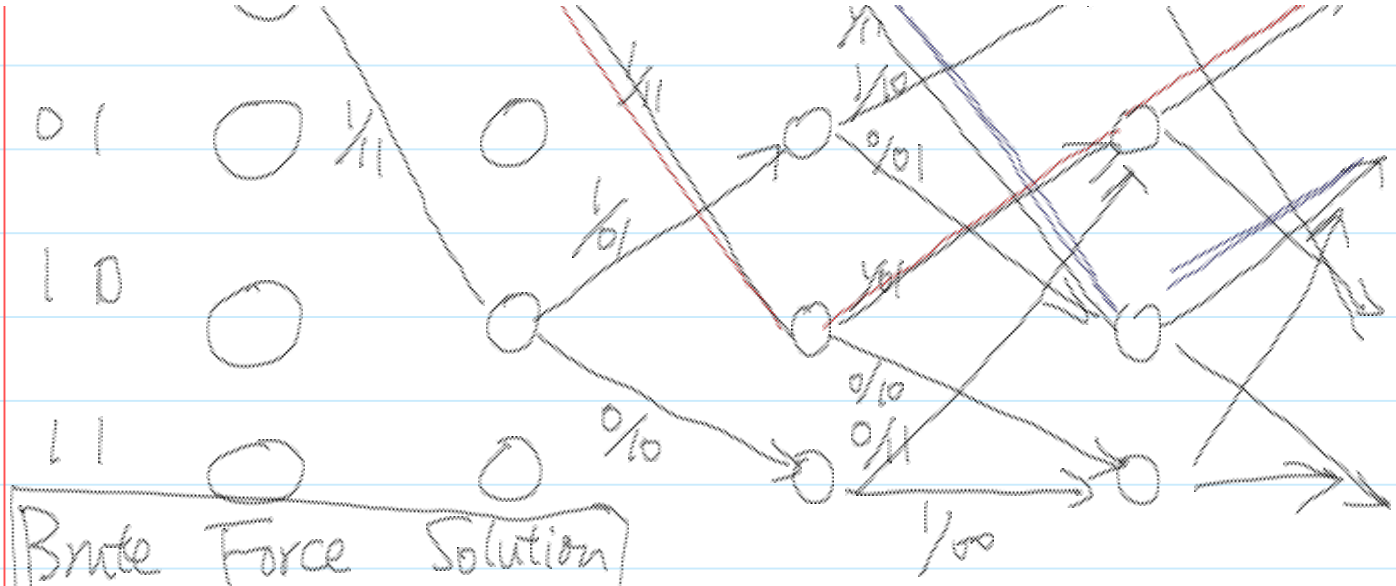
Example: we run the encoder for $N=4$ time slots.

$\Rightarrow$ info bits: $u_1\ u_2\ u_3\ u_4$

coded bits $v_1v_2\ v_3v_4\ \cdots\ v_7v_8$

$\vec{x}_{011} = 00\ 11\ 01\ 10$ $\vec{x}_{001} = 00\ 00\ 11\ 01$





Brute Force Solution

Repeat the 16 - Hypotheses testing

Find $\underset{\vec{x}_m}{\operatorname{argmax}} P_{Y|\vec{X}}(\vec{y}|\vec{x}_m)$ the ML decoder

The Viterbi Algorithm
1967

An efficient ML decoder for convolutional codes

Assume an i.i.d. channel.

Given any $\vec{y}_{obs}$, for any codeword $\vec{x}_m$ or for any given path p in the trellis, the likelihood value is

$P_{y_1y_2|v_1v_3} \cdot P_{y_3y_4|v_3v_4} \cdot P_{y_5y_6|v_5v_6} \cdot P_{y_7y_8|v_7v_8}$

if we let $\overline{R[1]}$ denote the output v_1v_2 of the first segment. $\overline{R[i]}$ denote $v_{2i-1}v_{2i}$

$$= f_1(\overline{R[1]})\, f_2(\overline{R[2]})\, f_3(\overline{R[3]})\, f_4(\overline{R[4]})$$

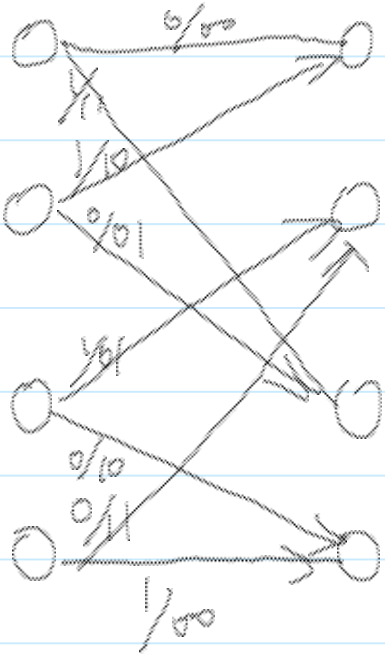
where $f_i(\cdot)$ is a $\{0,1\}^2 \mapsto [0,1]$ function that is constructed by the observation $y_{2i-1}\, y_{2i}$

That is, given the observation $y_1 \cdots y_8$ the functions $f_1(\cdot)$ to $f_4(\cdot)$ are decided by the likelihood $P(y_{2i-1}y_{2i} \mid \cdot, \cdot)$

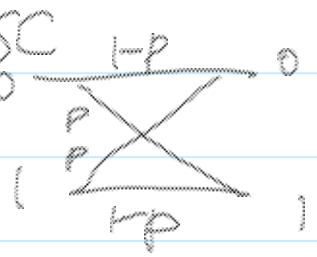
The goal is to find a path $\overline{R^*} = (\overline{R[1]}, \overline{R[2]}, \overline{R[3]}, \overline{R[4]})$ that

maximizes

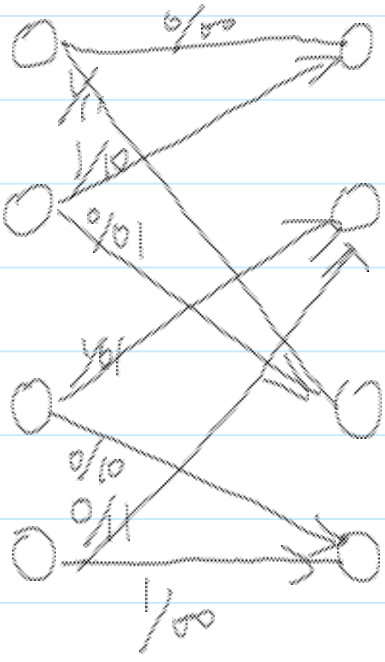
$$f_1(p[1])\, f_2(p[2])\, f_3(p[3])\, f_4(p[4])$$



ex: if BSC

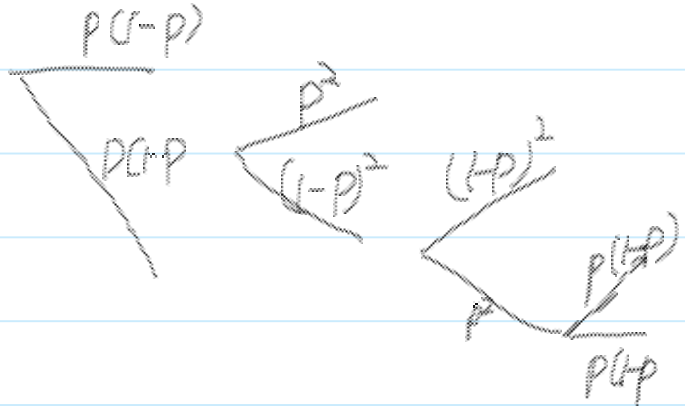


then



if $y_{2t-1}, y_{2t} = 0, 1$

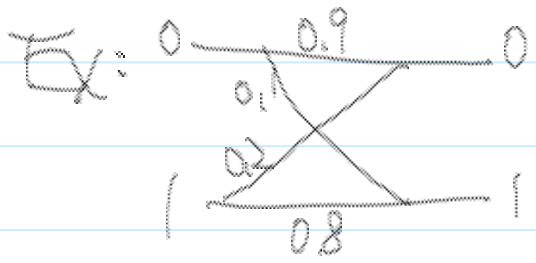
then



For different channel models & different observations, the partial objective function is different.

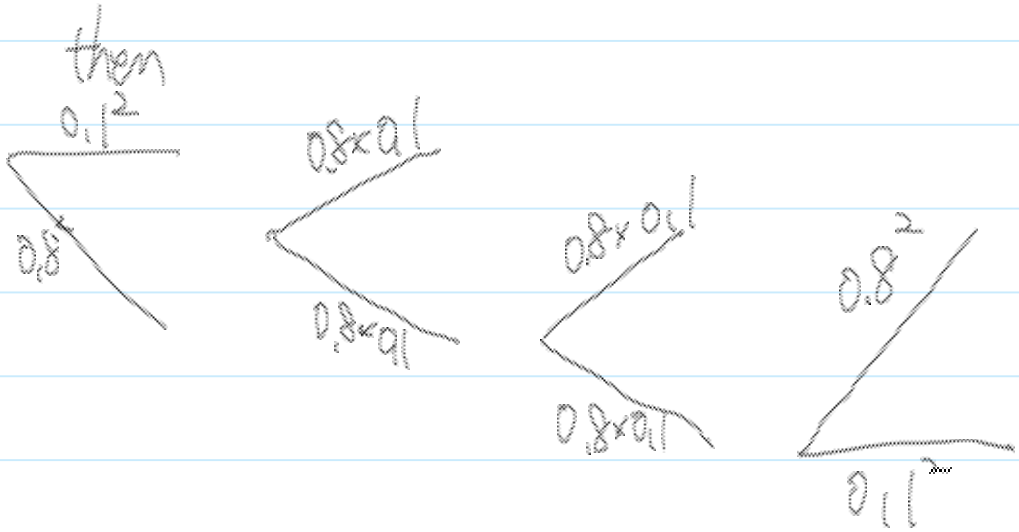
partial objective function is different.

Ex:



$y_{st-1}, y_{st} = 1,1$

then



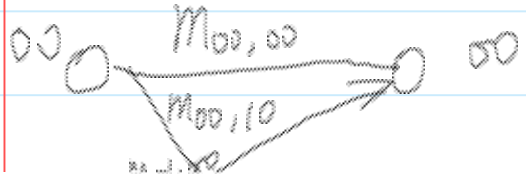
Ex: $P_{Y|X}(\cdot|x) \sim GM(\{-1\}^n, \sigma^2)$

$y_{st-1}, y_{st} = (0.7, -\sqrt{2})$

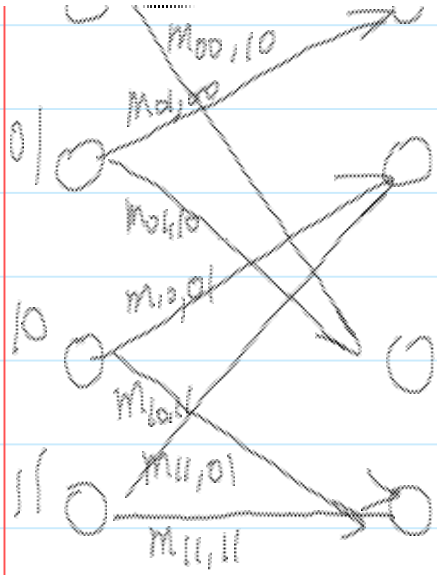
Then

$$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(0.7-1)^2}{2\sigma^2}} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(-\sqrt{2}-1)^2}{2\sigma^2}}$$

$$\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(0.7-(-1))^2}{2\sigma^2}} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(-\sqrt{2}-(-1))^2}{2\sigma^2}}$$



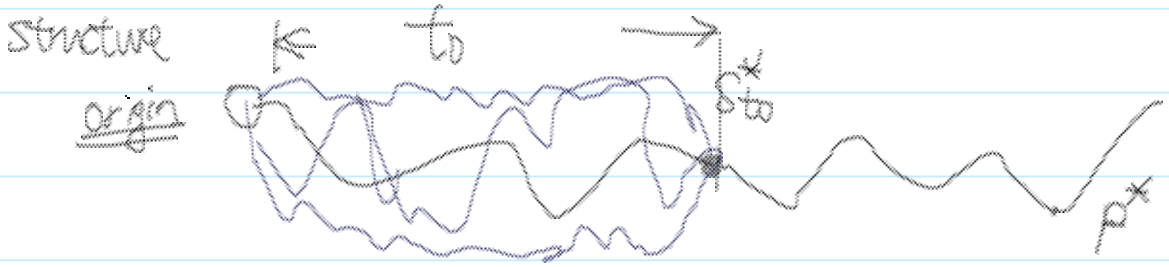
In the end, each



In the end, each segment has a metric $m_{s_t, s_{t+1}}$

* The VA algorithm

Observation: If a path p^* maximizes $\prod_{t=1}^{n} f_t(p[t])$ among all paths. let $s_{t_0}^*$ denote the state that p^* is using at the t_0-th stage of the trellis



then the partial path $0 \xrightarrow{p^*} s_{t_0}^*$ is

also the partial path maximizing

$$\prod_{t=1}^{t_0} f_t(R[t])$$

among all partial paths from 0 to $S_{t_0}^{*}$

(This observation is also the basis of the Dijkstra shortest path algorithm & the dynamic programming.)