Sequence Models

K Sri Rama Murty

IIT Hyderabad

ksrm@ee.iith.ac.in

April 14, 2022

Issues with Sequence Matching

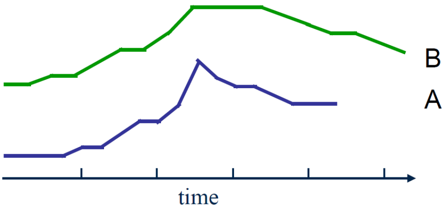
- Sequential data from a natural process may not only vary in magnitude but also in time-evaluation of the process

Issues with Sequence Matching

- Sequential data from a natural process may not only vary in magnitude but also in time-evaluation of the process
- Two realizations of the same process may differ in length
  - The rate of speech may slightly vary between repetitions
  - Time taken to complete an action or gesture
  - Speech recognition, text-dependent speaker recognition, spell-checker, sentence recommendation, video action recognition

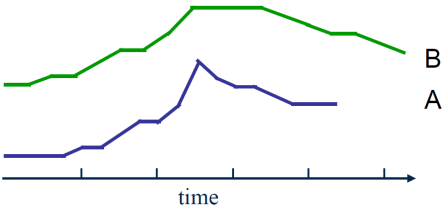
Issues with Sequence Matching

- Sequential data from a natural process may not only vary in magnitude but also in time-evaluation of the process
- Two realizations of the same process may differ in length
  - The rate of speech may slightly vary between repetitions
  - Time taken to complete an action or gesture
  - Speech recognition, text-dependent speaker recognition, spell-checker, sentence recommendation, video action recognition



Issues with Sequence Matching

- Sequential data from a natural process may not only vary in magnitude but also in time-evaluation of the process
- Two realizations of the same process may differ in length
  - The rate of speech may slightly vary between repetitions
  - Time taken to complete an action or gesture
  - Speech recognition, text-dependent speaker recognition, spell-checker, sentence recommendation, video action recognition



- How to compute distance between two sequences D(A,B)?

If Sequences are of Same Length!

- If sequences A and B are of same length, can we compute distance between corresponding time-instants?

If Sequences are of Same Length!

- If sequences A and B are of same length, can we compute distance between corresponding time-instants?

$$A = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N) \qquad B = (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_N)$$

$$D(A, B) = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{x}_n - \mathbf{y}_n\|$$

- Such a strategy may not work for misaligned sequences

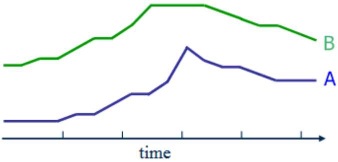
If Sequences are of Same Length!

- If sequences A and B are of same length, can we compute distance between corresponding time-instants?

$$A = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N) \qquad B = (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_N)$$

$$D(A, B) = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{x}_n - \mathbf{y}_n\|$$

- Such a strategy may not work for misaligned sequences

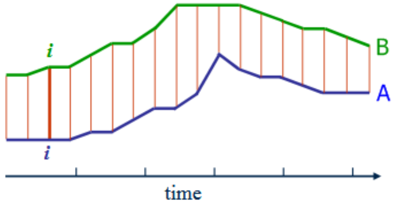


Motivation for Dynamic Time Warping

- Distance between corresponding points produce a poor similarity

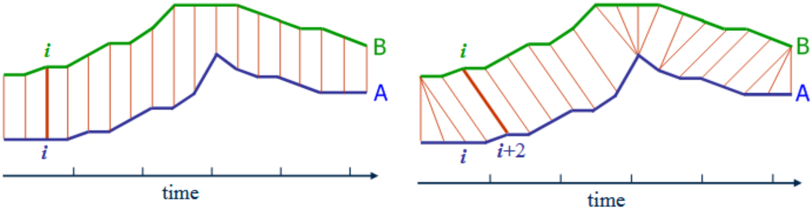
Motivation for Dynamic Time Warping

- Distance between corresponding points produce a poor similarity



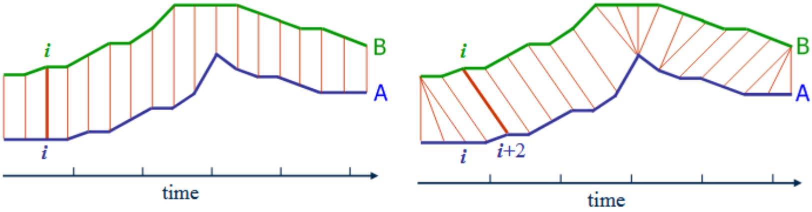
Motivation for Dynamic Time Warping

- Distance between corresponding points produce a poor similarity



Motivation for Dynamic Time Warping

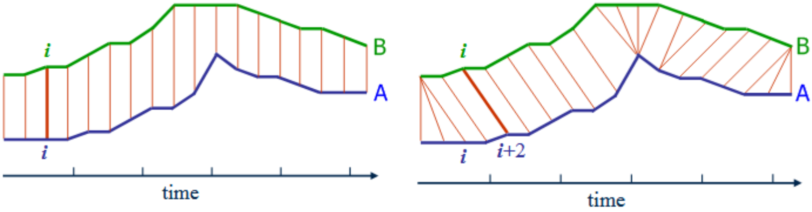
- Distance between corresponding points produce a poor similarity



- Nonlinear (elastic) alignment produces a more intuitive similarity measure

Motivation for Dynamic Time Warping

- Distance between corresponding points produce a poor similarity



- Nonlinear (elastic) alignment produces a more intuitive similarity measure
- Allows similar shapes to match even if they are out of phase in time

Distance between Sequences

Distance between Sequences

- Let A and B be two varying-length sequences

$$A = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M) \qquad B = (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_N)$$

Distance between Sequences

- Let A and B be two varying-length sequences

$$A = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M) \qquad B = (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_N)$$

- Find the corresponding points along best alignment path

$$(p_1, q_1), (p_2, q_2), \cdots (p_K, q_K) \qquad 1 \leq p_k \leq M \;\&\; 1 \leq q_k \leq N$$

Distance between Sequences

- Let A and B be two varying-length sequences

$$A = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M) \qquad B = (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_N)$$

- Find the corresponding points along best alignment path

$$(p_1, q_1), (p_2, q_2), \cdots (p_K, q_K) \qquad 1 \leq p_k \leq M \;\&\; 1 \leq q_k \leq N$$

- Distance between the sequences can be computed as

$$d(A, B) = \frac{1}{K} \sum_{k=1}^{K} \|\mathbf{x}_{p_k} - \mathbf{y}_{q_k}\|$$

Distance between Sequences

- Let A and B be two varying-length sequences

$$A = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M) \qquad B = (\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_N)$$

- Find the corresponding points along best alignment path

$$(p_1, q_1), (p_2, q_2), \cdots (p_K, q_K) \qquad 1 \leq p_k \leq M \;\&\; 1 \leq q_k \leq N$$

- Distance between the sequences can be computed as

$$d(A, B) = \frac{1}{K} \sum_{k=1}^{K} \|\mathbf{x}_{p_k} - \mathbf{y}_{q_k}\|$$

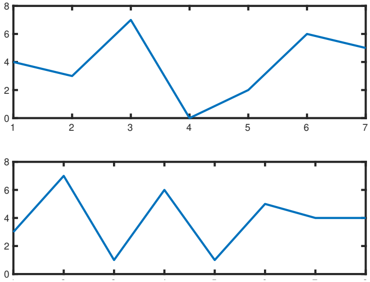
- Need to follow constraints while identifying corresponding points

Distance Matrix

$$A = (4, 3, 7, 0, 2, 6, 5)$$
$$B = (3, 7, 1, 6, 1, 5, 4, 4)$$

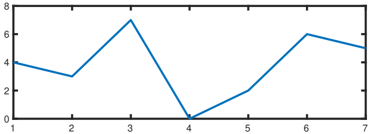
Distance Matrix

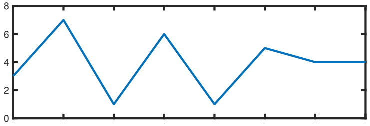
$$A = (4, 3, 7, 0, 2, 6, 5)$$
$$B = (3, 7, 1, 6, 1, 5, 4, 4)$$



Distance Matrix

$$A = (4, 3, 7, 0, 2, 6, 5)$$
$$B = (3, 7, 1, 6, 1, 5, 4, 4)$$



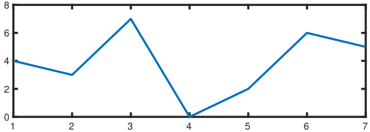


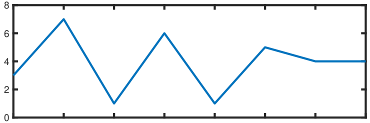
- Distance matrix

$$\mathbf{D}(i, j) = \|x_i - y_j\|$$

Distance Matrix

$$A = (4, 3, 7, 0, 2, 6, 5)$$
$$B = (3, 7, 1, 6, 1, 5, 4, 4)$$





- Distance matrix

$$\mathbf{D}(i, j) = \|x_i - y_j\|$$

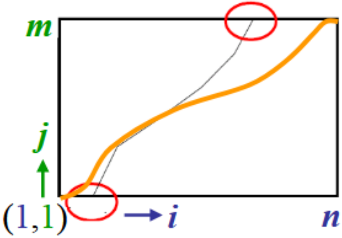
4	0	1	9	16	4	4	1
4	0	1	9	16	4	4	1
5	1	4	4	25	9	1	0
1	9	4	36	1	1	25	16
6	4	9	1	36	16	0	1
1	9	4	36	1	1	25	16
7	9	16	0	49	25	1	4
3	1	0	16	9	1	9	4
	4	3	7	0	2	6	5

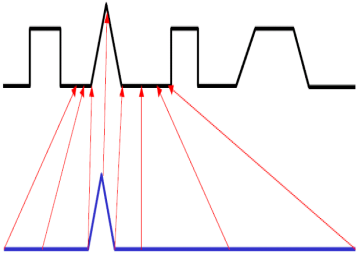
Boundary Condition

- The alignment path should start at bottom-left and end at top-right.

$$(p_1, q_1) = (1, 1) \qquad (p_K, q_K) = (M, N)$$

- Ensures that the alignment is not limited to a partial sequence



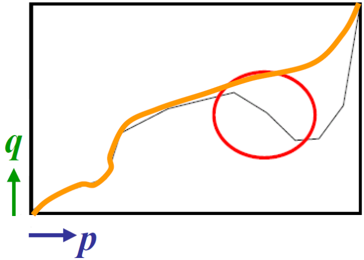


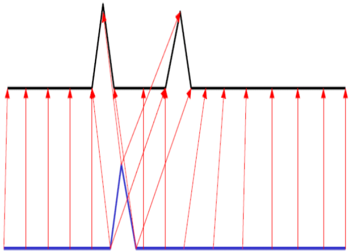
Monotonicity

- The alignment path should not go back in 'time' index.

$$p_{k-1} \leq p_k \qquad q_{k-1} \leq q_k$$

- Ensures that the features are not repeated in the alignment



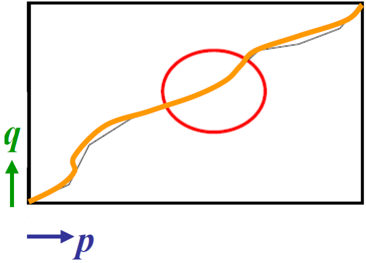


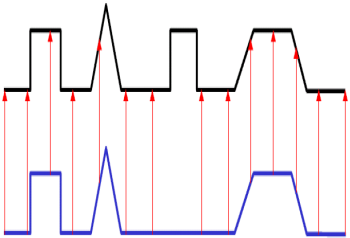
Continuity

- The alignment path should not jump in 'time' index.

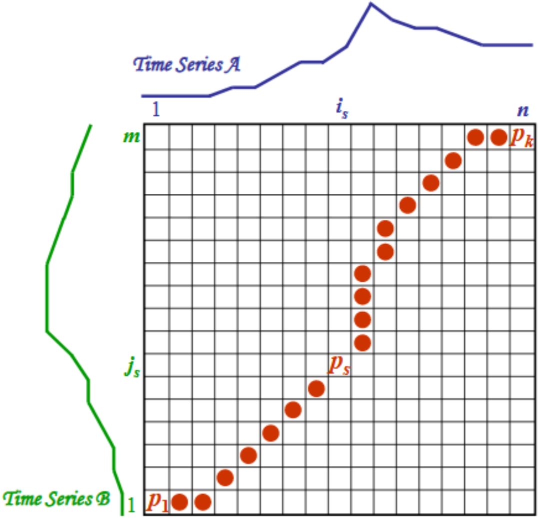
$$p_k - p_{k-1} \leq 1 \qquad q_k - q_{k-1} \leq 1$$

- Ensures that the alignment path does not omit important features





DTW Alignment Path



Determining Alignment Path

$$A = (4, 3, 7, 0, 2, 6, 5) \qquad B = (3, 7, 1, 6, 1, 5, 4, 4)$$

Determining Alignment Path

$$A = (4, 3, 7, 0, 2, 6, 5) \qquad B = (3, 7, 1, 6, 1, 5, 4, 4)$$

8	4	0	1	9	16	4	4	1
7	4	0	1	9	16	4	4	1
6	5	1	4	4	25	9	1	0
5	1	9	4	36	1	1	25	16
4	6	4	9	1	36	16	0	1
3	1	9	4	36	1	1	25	16
2	7	9	16	0	49	25	1	4
1	3	1	0	16	9	1	9	4
		4	3	7	0	2	6	5
		1	2	3	4	5	6	7

Determining Alignment Path

$$A = (4, 3, 7, 0, 2, 6, 5) \qquad B = (3, 7, 1, 6, 1, 5, 4, 4)$$

Determining Alignment Path

$$A = (4, 3, 7, 0, 2, 6, 5) \qquad B = (3, 7, 1, 6, 1, 5, 4, 4)$$

8	4	33 0	33 1	41 9	56 16	33 4	26 4	20(7,7) 1
7	4	33 0	32 1	40 9	47 16	29 4	22 4	19(6,6) 1
6	5	33 1	31 4	31 4	41 25	25 9	18(5,5) 1	18(1,1) 0
5	1	32 9	27 4	51 36	16(3,4) 1	17(4,5) 1	28 25	19 16
4	6	23 4	23 9	15(2,3) 1	38 36	18 16	3(5,3) 0	4(6,4) 1
3	1	19 9	14(1,2) 4	37 36	2(3,2) 1	3(4,3) 1	28 25	44 16
2	7	10(1,1) 9	17(1,1) 16	1(2,2) 0	50(3,2) 49	51(4,1) 25	28(5,1) 1	32(6,2) 4
1	3	1 1	1(1,1) 0	17(2,1) 16	26(3,1) 9	27(4,1) 1	36(5,1) 9	40(6,1) 4
		4	3	7	0	2	6	5
		1	2	3	4	5	6	7

DTW Algorithm

DTW Algorithm

- Compute distance matrix $\mathbf{D}_{M \times N}$ between A and B

DTW Algorithm

- Compute distance matrix $\mathbf{D}_{M \times N}$ between A and B
- Compute Accumulated distance matrix $\mathbf{C}$

DTW Algorithm

- Compute distance matrix $\mathbf{D}_{M \times N}$ between A and B
- Compute Accumulated distance matrix $\mathbf{C}$
  - Initialization: $\mathbf{C}(0,0) = \mathbf{D}(0,0)$

DTW Algorithm

- Compute distance matrix $\mathbf{D}_{M \times N}$ between A and B
- Compute Accumulated distance matrix $\mathbf{C}$
  - Initialization: $\mathbf{C}(0,0) = \mathbf{D}(0,0)$
  - For $i = 1, 2, \cdots, M$ and $j = 1, 2, \cdots, N$

DTW Algorithm

- Compute distance matrix $\mathbf{D}_{M \times N}$ between A and B
- Compute Accumulated distance matrix $\mathbf{C}$
  - Initialization: $\mathbf{C}(0,0) = \mathbf{D}(0,0)$
  - For $i = 1, 2, \cdots, M$ and $j = 1, 2, \cdots, N$
    - Update the accumulated distance matrix

$$\mathbf{C}(i,j) = \mathbf{D}(i,j) + \min\{\mathbf{C}(i-1,j-1),\ \mathbf{C}(i,j-1),\ \mathbf{C}(i-1,j)\}$$

DTW Algorithm

- Compute distance matrix $\mathbf{D}_{M\times N}$ between A and B
- Compute Accumulated distance matrix $\mathbf{C}$
  - Initialization: $\mathbf{C}(0,0) = \mathbf{D}(0,0)$
  - For $i = 1, 2, \cdots, M$ and $j = 1, 2, \cdots, N$
    - Update the accumulated distance matrix

$$\mathbf{C}(i,j) = \mathbf{D}(i,j) + \min\{\mathbf{C}(i-1,j-1),\ \mathbf{C}(i,j-1),\ \mathbf{C}(i-1,j)\}$$

    - Store the best path transition

$$\mathbf{P}(i,j) = \arg\min\{\mathbf{C}(i-1,j-1),\ \mathbf{C}(i,j-1),\ \mathbf{C}(i-1,j)\}$$

DTW Algorithm

- Compute distance matrix $\mathbf{D}_{M \times N}$ between A and B
- Compute Accumulated distance matrix $\mathbf{C}$
  - Initialization: $\mathbf{C}(0,0) = \mathbf{D}(0,0)$
  - For $i = 1, 2, \cdots, M$ and $j = 1, 2, \cdots, N$
    - Update the accumulated distance matrix

$$\mathbf{C}(i,j) = \mathbf{D}(i,j) + \min\{\mathbf{C}(i-1, j-1),\ \mathbf{C}(i, j-1),\ \mathbf{C}(i-1, j)\}$$

    - Store the best path transition

$$\mathbf{P}(i,j) = \arg\min\{\mathbf{C}(i-1, j-1),\ \mathbf{C}(i, j-1),\ \mathbf{C}(i-1, j)\}$$

- $\mathbf{C}(M, N)$ indicates the similarity measure between the sequences

DTW Algorithm

- Compute distance matrix $\mathbf{D}_{M\times N}$ between A and B
- Compute Accumulated distance matrix $\mathbf{C}$
  - Initialization: $\mathbf{C}(0,0) = \mathbf{D}(0,0)$
  - For $i = 1, 2, \cdots, M$ and $j = 1, 2, \cdots, N$
    - Update the accumulated distance matrix

$$\mathbf{C}(i,j) = \mathbf{D}(i,j) + \min\{\mathbf{C}(i-1,j-1),\ \mathbf{C}(i,j-1),\ \mathbf{C}(i-1,j)\}$$

    - Store the best path transition

$$\mathbf{P}(i,j) = \arg\min\{\mathbf{C}(i-1,j-1),\ \mathbf{C}(i,j-1),\ \mathbf{C}(i-1,j)\}$$

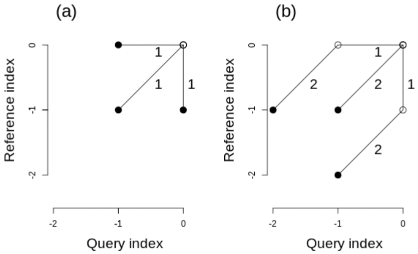
- $\mathbf{C}(M,N)$ indicates the similarity measure between the sequences
- Best possible alignment can be obtained by backtracking $P(i,j)$

$$q_T^* = (M,N) \qquad q_{t-1}^* = \mathbf{P}(q_t^*) \qquad t = T, T-1, \cdots 1$$

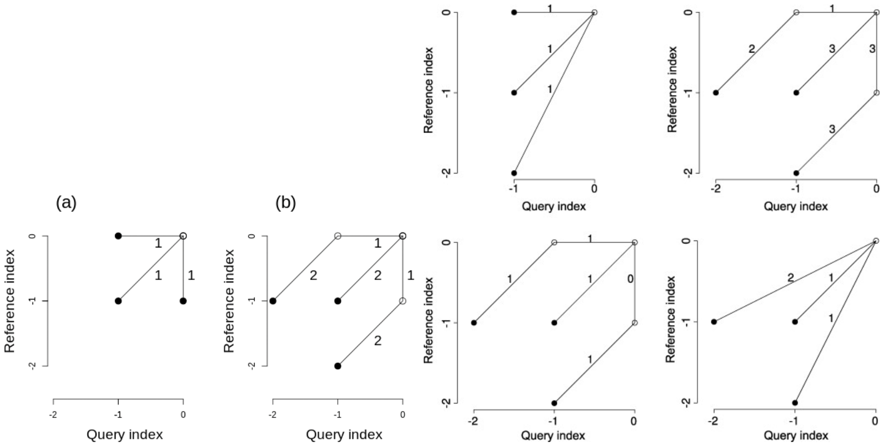
Summary of DTW

- DTW - quantifies distance between varying-length sequences
- Computational complexity: $O(NM)$
- DTW is not a distance metric
  - DTW is not commutative: $D(A, B) \neq D(B, A)$ for asymmetric steps
  - Does not satisfy triangular inequality
- DTW is a template-matching technique
  - Does not account for statistical variations
- Statistical models - Markov models and hidden Markov models

Symmetric vs Asymmetric Step Patterns



Symmetric vs Asymmetric Step Patterns



Weather Prediction

Weather Prediction

- What will be tomorrow's weather (Sunny, Cloudy or Rainy)?

Weather Prediction

- What will be tomorrow's weather (Sunny, Cloudy or Rainy)?
  - Can we predict it by considering a PMF over S, C and R?

Weather Prediction

- What will be tomorrow's weather (Sunny, Cloudy or Rainy)?
  - Can we predict it by considering a PMF over S, C and R?
  - Past history is important in weather prediction.

Weather Prediction

- What will be tomorrow's weather (Sunny, Cloudy or Rainy)?
  - Can we predict it by considering a PMF over S, C and R?
  - Past history is important in weather prediction.
- Given the past three days weather, predict tomorrow's weather

$$P[q_4 = R/q_3 = C,\ q_2 = S,\ q_1 = S]$$

Weather Prediction

- What will be tomorrow's weather (Sunny, Cloudy or Rainy)?
  - Can we predict it by considering a PMF over S, C and R?
  - Past history is important in weather prediction.
- Given the past three days weather, predict tomorrow's weather

$$P[q_4 = R/q_3 = C,\ q_2 = S,\ q_1 = S]$$

  - How to estimate this quantity? Given news papers for the last one year!

Weather Prediction

- What will be tomorrow's weather (Sunny, Cloudy or Rainy)?
  - Can we predict it by considering a PMF over S, C and R?
  - Past history is important in weather prediction.
- Given the past three days weather, predict tomorrow's weather

$$P[q_4 = R / q_3 = C, \; q_2 = S, \; q_1 = S]$$

  - How to estimate this quantity? Given news papers for the last one year!
  - Need to collect all instances of 'RCSS' from the paper.

Weather Prediction

- What will be tomorrow's weather (Sunny, Cloudy or Rainy)?
  - Can we predict it by considering a PMF over S, C and R?
  - Past history is important in weather prediction.
- Given the past three days weather, predict tomorrow's weather

$$P[q_4 = R / q_3 = C, \; q_2 = S, \; q_1 = S]$$

  - How to estimate this quantity? Given news papers for the last one year!
  - Need to collect all instances of 'RCSS' from the paper.
  - In general for a T day history, there will be 3^T possibilities.

Weather Prediction

- What will be tomorrow's weather (Sunny, Cloudy or Rainy)?
  - Can we predict it by considering a PMF over S, C and R?
  - Past history is important in weather prediction.
- Given the past three days weather, predict tomorrow's weather

$$P[q_4 = R / q_3 = C, \; q_2 = S, \; q_1 = S]$$

  - How to estimate this quantity? Given news papers for the last one year!
  - Need to collect all instances of 'RCSS' from the paper.
  - In general for a T day history, there will be 3^T possibilities.
  - Most of those weather conditions may not have occurred even once!

Weather Prediction

- What will be tomorrow's weather (Sunny, Cloudy or Rainy)?
  - Can we predict it by considering a PMF over S, C and R?
  - Past history is important in weather prediction.
- Given the past three days weather, predict tomorrow's weather

$$P[q_4 = R / q_3 = C,\ q_2 = S,\ q_1 = S]$$

  - How to estimate this quantity? Given news papers for the last one year!
  - Need to collect all instances of 'RCSS' from the paper.
  - In general for a T day history, there will be 3^T possibilities.
  - Most of those weather conditions may not have occurred even once!
- Let us just restrict the memory to a single-step

Weather Prediction

- What will be tomorrow's weather (Sunny, Cloudy or Rainy)?
  - Can we predict it by considering a PMF over S, C and R?
  - Past history is important in weather prediction.
- Given the past three days weather, predict tomorrow's weather

$$P[q_4 = R/q_3 = C,\ q_2 = S,\ q_1 = S]$$

  - How to estimate this quantity? Given news papers for the last one year!
  - Need to collect all instances of 'RCSS' from the paper.
  - In general for a T day history, there will be 3^T possibilities.
  - Most of those weather conditions may not have occurred even once!
- Let us just restrict the memory to a single-step
  - Tomorrows weather depends only on today's weather, no matter what happened prior to that.

First Order Markov Model

First Order Markov Model

- Future depends only on present (and not on past)

$$p(q_n \ / \ q_{n-1}, \ q_{n-2}, \cdots, \ q_1) = p(q_n / q_{n-1})$$

q_ks can take any of the N discrete symbols, e.g. S, C or R

First Order Markov Model

- Future depends only on present (and not on past)

$$p(q_n \;/\; q_{n-1},\; q_{n-2}, \cdots,\; q_1) = p(q_n / q_{n-1})$$

q_ks can take any of the N discrete symbols, e.g. S, C or R

- As a consequence, the joint density can be expressed as

$$p(q_n, q_{n-1}, \cdots, q_2, q_1) = p(q_0) \prod_{k=1}^{n} p(q_k / q_{k-1})$$

First Order Markov Model

- Future depends only on present (and not on past)

$$p(q_n \ / \ q_{n-1}, \ q_{n-2}, \cdots, \ q_1) = p(q_n/q_{n-1})$$

q_ks can take any of the N discrete symbols, e.g. S, C or R

- As a consequence, the joint density can be expressed as

$$p(q_n, q_{n-1}, \cdots, q_2, q_1) = p(q_0) \prod_{k=1}^{n} p(q_k/q_{k-1})$$

- First order Markov model can be completely described by

First Order Markov Model

- Future depends only on present (and not on past)

$$p(q_n \;/\; q_{n-1},\; q_{n-2}, \cdots,\; q_1) = p(q_n / q_{n-1})$$

  q_ks can take any of the N discrete symbols, e.g. S, C or R

- As a consequence, the joint density can be expressed as

$$p(q_n, q_{n-1}, \cdots, q_2, q_1) = p(q_0) \prod_{k=1}^{n} p(q_k / q_{k-1})$$

- First order Markov model can be completely described by
  - N - Number of distinct states (or symbols) of the model

First Order Markov Model

- Future depends only on present (and not on past)

$$p(q_n \;/\; q_{n-1},\; q_{n-2}, \cdots,\; q_1) = p(q_n/q_{n-1})$$

q_ks can take any of the N discrete symbols, e.g. S, C or R

- As a consequence, the joint density can be expressed as

$$p(q_n, q_{n-1}, \cdots, q_2, q_1) = p(q_0) \prod_{k=1}^{n} p(q_k/q_{k-1})$$

- First order Markov model can be completely described by
  - N - Number of distinct states (or symbols) of the model
  - π_k - Initial probability of starting in state k at time $t = 0$

First Order Markov Model

- Future depends only on present (and not on past)

$$p(q_n \ / \ q_{n-1}, \ q_{n-2}, \cdots, \ q_1) = p(q_n/q_{n-1})$$

  q_ks can take any of the N discrete symbols, e.g. S, C or R

- As a consequence, the joint density can be expressed as

$$p(q_n, q_{n-1}, \cdots, q_2, q_1) = p(q_0) \prod_{k=1}^{n} p(q_k/q_{k-1})$$

- First order Markov model can be completely described by
  - N - Number of distinct states (or symbols) of the model
  - π_k - Initial probability of starting in state k at time $t = 0$
  - $\mathbf{A}_{N \times N} = [a_{ij}]$: State transition probability of moving from state s_i to state s_j

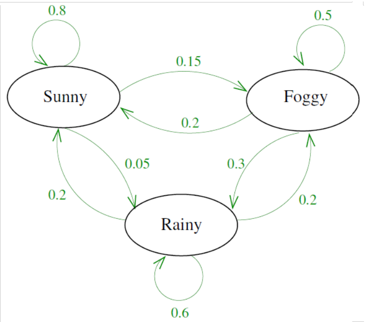
Markov Model for Weather Prediction

Markov Model for Weather Prediction

Tod $\longrightarrow$ Tom	S	C	R
Sunny	0.8	0.15	0.05

Markov Model for Weather Prediction

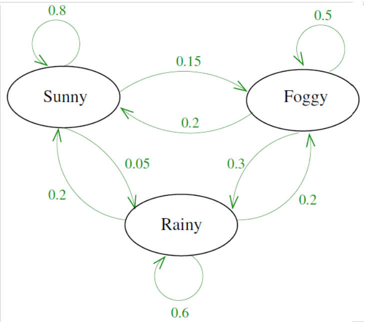
Tod $\longrightarrow$ Tom	S	C	R
Sunny	0.8	0.15	0.05
Cloudy	0.2	0.5	0.3
Rainy	0.2	0.2	0.6



$$P[q_4 = R/q_3 = C,\ q_2 = S,\ q_1 = S] =$$

Markov Model for Weather Prediction

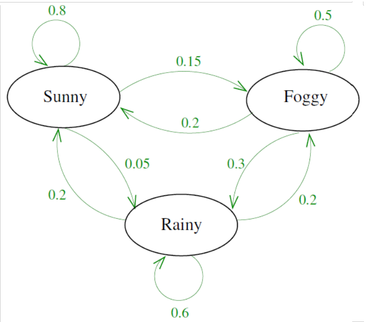
Tod $\longrightarrow$ Tom	S	C	R
Sunny	0.8	0.15	0.05
Cloudy	0.2	0.5	0.3
Rainy	0.2	0.2	0.6



$$P[q_4 = R/q_3 = C,\ q_2 = S,\ q_1 = S] = P[q_4 = R/q_3 = C]$$

Markov Model for Weather Prediction

Tod $\longrightarrow$ Tom	S	C	R
Sunny	0.8	0.15	0.05
Cloudy	0.2	0.5	0.3
Rainy	0.2	0.2	0.6

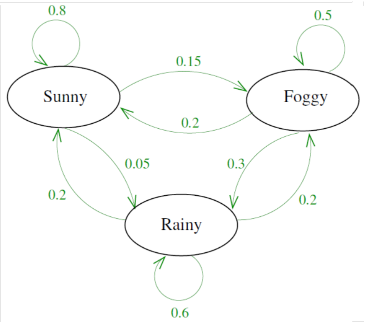


$$P[q_4 = R/q_3 = C,\ q_2 = S,\ q_1 = S] = P[q_4 = R/q_3 = C] = 0.3$$

$$P[q_2 = S, q_3 = R/q_1 = S] \quad =$$

Markov Model for Weather Prediction

Tod $\longrightarrow$ Tom	S	C	R
Sunny	0.8	0.15	0.05
Cloudy	0.2	0.5	0.3
Rainy	0.2	0.2	0.6



$$P[q_4 = R/q_3 = C,\ q_2 = S,\ q_1 = S] = P[q_4 = R/q_3 = C] = 0.3$$

$$\begin{aligned} P[q_2 = S, q_3 = R/q_1 = S] &= P[q_2 = S/q_1 = S]P[q_3 = R/q_2 = S] \\ &= 0.8 \times 0.05 = 0.04 \end{aligned}$$

Probabilistic Language Models

Probabilistic Language Models

- Assign probability to a sentence

$$p(\text{Sentence}) = p(w_1, w_2, w_3, \cdots w_N)$$

Probabilistic Language Models

- Assign probability to a sentence

$$p(\text{Sentence}) = p(w_1, w_2, w_3, \cdots w_N)$$

- Speech recognition

Probabilistic Language Models

- Assign probability to a sentence

$$p(\text{Sentence}) = p(w_1, w_2, w_3, \cdots w_N)$$

- Speech recognition
  - p(I saw a van) ¿ p(eyes awe of an)

Probabilistic Language Models

- Assign probability to a sentence

$$p(\text{Sentence}) = p(w_1, w_2, w_3, \cdots w_N)$$

- Speech recognition
  - p(I saw a van) ¿ p(eyes awe of an)
  - It's a nice hockey match vs It's an ice hockey match

Probabilistic Language Models

- Assign probability to a sentence

$$p(\text{Sentence}) = p(w_1, w_2, w_3, \cdots w_N)$$

- Speech recognition
  - p(I saw a van) ¿ p(eyes awe of an)
  - It's a nice hockey match vs It's an ice hockey match
- Spell Correction

$$p(\text{about fifteen minutes from}) > p(\text{aboufifteen minutes from})$$

Probabilistic Language Models

- Assign probability to a sentence

$$p(\text{Sentence}) = p(w_1, w_2, w_3, \cdots w_N)$$

- Speech recognition
  - p(I saw a van) ¿ p(eyes awe of an)
  - It's a nice hockey match vs It's an ice hockey match
- Spell Correction

$$p(\text{about fifteen minutes from}) > p(\text{aboufifteen minutes from})$$

- Machine translation

$$p(\text{high winds tonight}) > p(\text{large winds tonight})$$

Probabilistic Language Models

- Assign probability to a sentence

$$p(\text{Sentence}) = p(w_1, w_2, w_3, \cdots w_N)$$

- Speech recognition
  - p(I saw a van) ¿ p(eyes awe of an)
  - It's a nice hockey match vs It's an ice hockey match
- Spell Correction

$$p(\text{about fifteen minutes from}) > p(\text{aboufifteen minutes from})$$

- Machine translation

$$p(\text{high winds tonight}) > p(\text{large winds tonight})$$

- Spoken dialogue systems, summerization etc.

Unigram Model

Unigram Model

- Assuming words in a sentence are independent

$$p(\mathbf{W}) = p(w_1, w_2, \cdots w_N) \approx \prod_{n=1}^{N} p(w_n)$$

Unigram Model

- Assuming words in a sentence are independent

$$p(\mathbf{W}) = p(w_1, w_2, \cdots w_N) \approx \prod_{n=1}^{N} p(w_n)$$

- Example sentences generated from unigram model

Unigram Model

- Assuming words in a sentence are independent

$$p(\mathbf{W}) = p(w_1, w_2, \cdots w_N) \approx \prod_{n=1}^{N} p(w_n)$$

- Example sentences generated from unigram model
  - young you wall last but and had in after n't words 'nothing more away here these In up ; man the door be up apathy one must relaxation. No papers many uneasiness : her kitchen

Unigram Model

- Assuming words in a sentence are independent

$$p(\mathbf{W}) = p(w_1, w_2, \cdots w_N) \approx \prod_{n=1}^{N} p(w_n)$$

- Example sentences generated from unigram model
  - young you wall last but and had in after n't words 'nothing more away here these In up ; man the door be up apathy one must relaxation. No papers many uneasiness : her kitchen
  - fifth an of futures the an incorporated a a the inflation most dollars quarter in is mass

Bigram Model (First Order Markov)

Bigram Model (First Order Markov)

- Condition the current word on the previous word

$$p(\mathbf{W}) \approx \prod_{n=1}^{N} p(w_n / w_{n-1})$$

Bigram Model (First Order Markov)

- Condition the current word on the previous word

$$p(\mathbf{W}) \approx \prod_{n=1}^{N} p(w_n / w_{n-1})$$

- ML estimate of bigram probabilities is given by

$$p(w_i / w_j) = \frac{C(w_j, w_i)}{C(w_j)}$$

Bigram Model (First Order Markov)

- Condition the current word on the previous word

$$p(\mathbf{W}) \approx \prod_{n=1}^{N} p(w_n / w_{n-1})$$

- ML estimate of bigram probabilities is given by

$$p(w_i / w_j) = \frac{C(w_j, w_i)}{C(w_j)}$$

- An example

Bigram Model (First Order Markov)

- Condition the current word on the previous word

$$p(\mathbf{W}) \approx \prod_{n=1}^{N} p(w_n / w_{n-1})$$

- ML estimate of bigram probabilities is given by

$$p(w_i / w_j) = \frac{C(w_j, w_i)}{C(w_j)}$$

- An example
  - (I am sam)

Bigram Model (First Order Markov)

- Condition the current word on the previous word

$$p(\mathbf{W}) \approx \prod_{n=1}^{N} p(w_n / w_{n-1})$$

- ML estimate of bigram probabilities is given by

$$p(w_i / w_j) = \frac{C(w_j, w_i)}{C(w_j)}$$

- An example
  - (I am sam)
  - (Sam I am)

Bigram Model (First Order Markov)

- Condition the current word on the previous word

$$p(\mathbf{W}) \approx \prod_{n=1}^{N} p(w_n/w_{n-1})$$

- ML estimate of bigram probabilities is given by

$$p(w_i/w_j) = \frac{C(w_j, w_i)}{C(w_j)}$$

- An example
  - (I am sam)
  - (Sam I am)
  - (I do not like green eggs and ham)

Bigram Model (First Order Markov)

- Condition the current word on the previous word

$$p(\mathbf{W}) \approx \prod_{n=1}^{N} p(w_n/w_{n-1})$$

- ML estimate of bigram probabilities is given by

$$p(w_i/w_j) = \frac{C(w_j, w_i)}{C(w_j)}$$

- An example
  - (I am sam)
  - (Sam I am)
  - (I do not like green eggs and ham)
  - Bigram probabilities:
    P(I/() = 2/3 P(Sam/()=1/3 P(am/I)=2/3 P()/Sam) = 1/2
    P(Sam/am) = 1/2 P(do/I)=1/3

Bigram Generated Sentences

- I must have taken into this way out of her by one hand, with which was one has been knocked off again
- outside new car parking lot of the agreement reached
- this would a record november

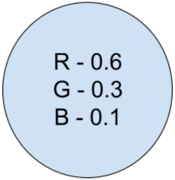
Summary of Markov Model

- First order Markov model can be completely described by
  - N - Number of distinct states (or symbols) of the model
  - π_k - Initial probability of starting in state k at time $t = 0$
  - $\mathbf{A}_{N \times N} = [a_{ij}]$: State transition probability of moving from state s_i to s_j

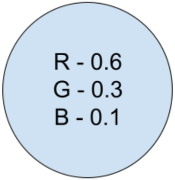
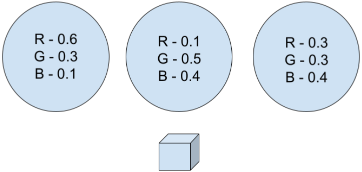
Summary of Markov Model

- First order Markov model can be completely described by
  - N - Number of distinct states (or symbols) of the model
  - π_k - Initial probability of starting in state k at time $t = 0$
  - $\mathbf{A}_{N \times N} = [a_{ij}]$: State transition probability of moving from state s_i to s_j
- State is deterministic in the case of Markov models
  - Each state produces a specific observation symbol
  - There is no difference between state and symbol
  - Given the observation symbol sequence, state sequence can be exactly determined

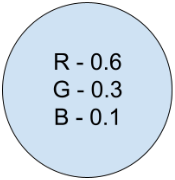
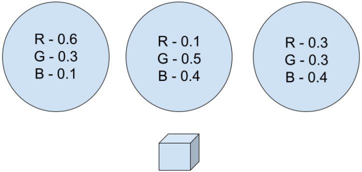
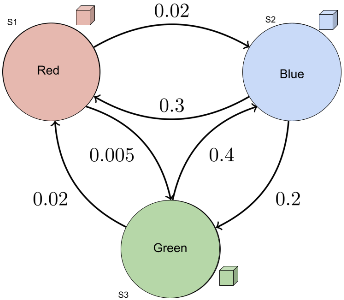
Basket Balls Experiment

	Deterministic State	Latent Variable
RV		

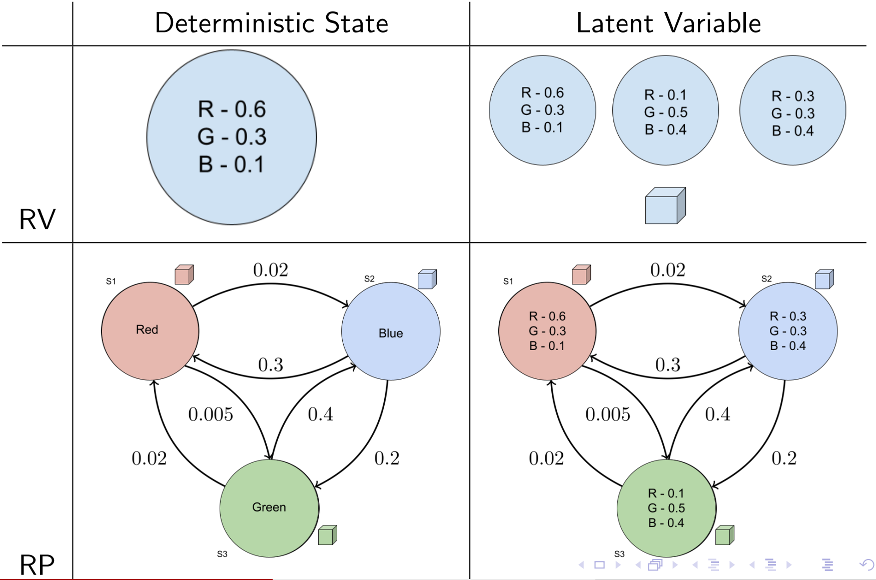
Basket Balls Experiment

	Deterministic State	Latent Variable
RV		

Basket Balls Experiment

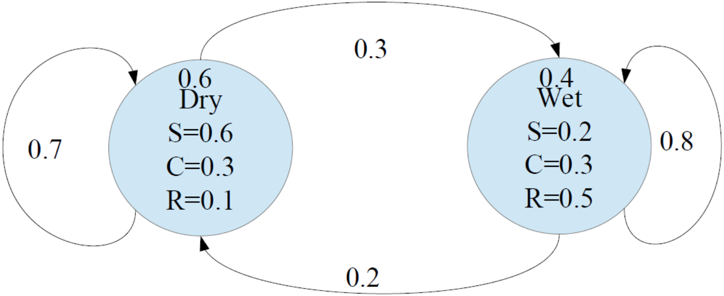
	Deterministic State	Latent Variable
RV		
RP		

Basket Balls Experiment

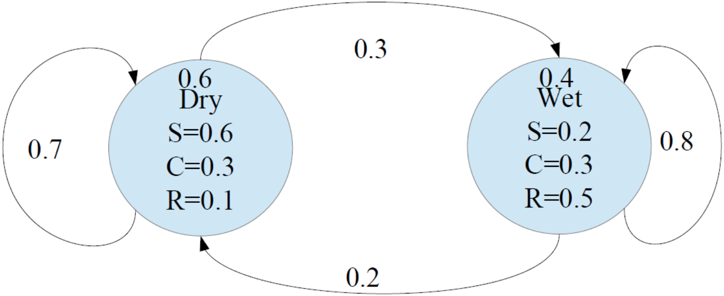


Hidden Markov Model

Hidden Markov Model

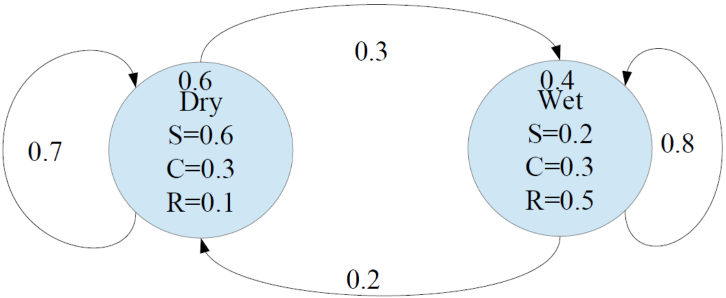


Hidden Markov Model



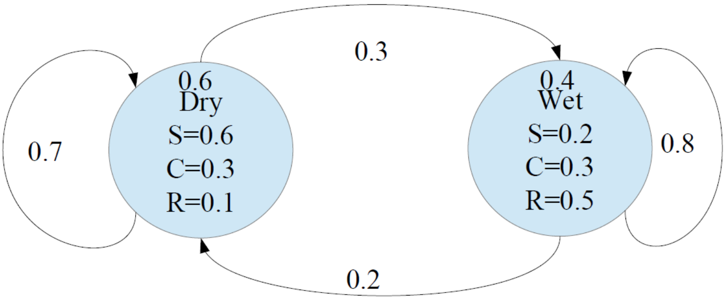
- State and symbol are different.

Hidden Markov Model



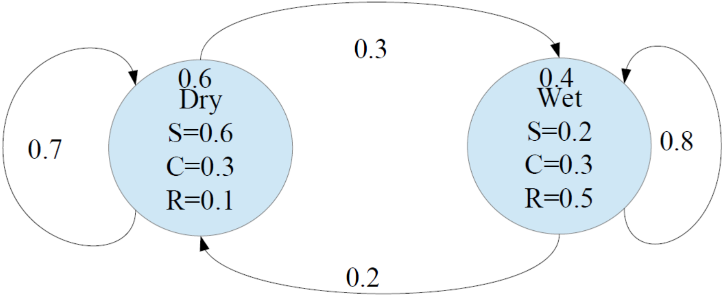
- State and symbol are different.
- All the states can produce the symbols with non-zero probability.

Hidden Markov Model



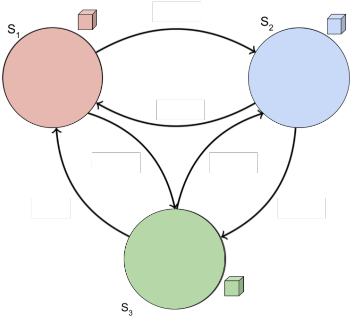
- State and symbol are different.
- All the states can produce the symbols with non-zero probability.
- Evaluate $P[o_2 = S, o_3 = R/Dry]$?

Hidden Markov Model

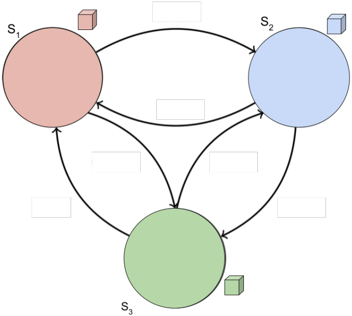


- State and symbol are different.
- All the states can produce the symbols with non-zero probability.
- Evaluate $P[o_2 = S, o_3 = R/Dry]$?
- Which state sequence produced the observation sequence?

HMM Parameters $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$



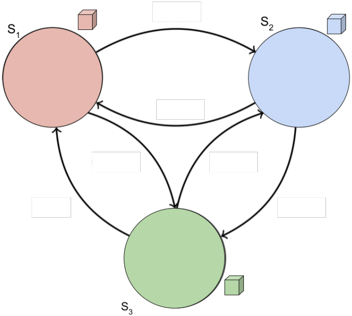
HMM Parameters $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$



- N - Number of states

$$s_1, s_2, \cdots, s_i, \cdots s_N$$

HMM Parameters $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$



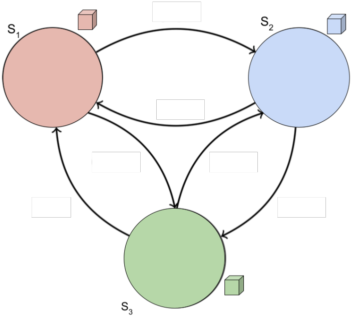
- N - Number of states

$$s_1, s_2, \cdots, s_i, \cdots s_N$$

- M - Number of symbols

$$v_1, v_2, \cdots, v_k, \cdots v_M$$

HMM Parameters $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$



- N - Number of states

$$s_1, s_2, \cdots, s_i, \cdots s_N$$

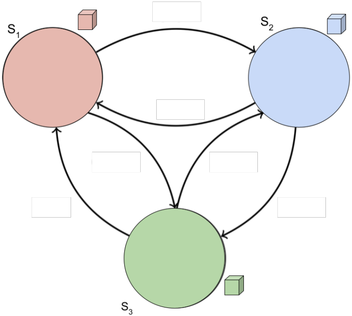
- M - Number of symbols

$$v_1, v_2, \cdots, v_k, \cdots v_M$$

- Initial state probability

$$\Pi = [\pi_i] = P[q_1 = s_i] \qquad 1 \leq i \leq N$$

HMM Parameters $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$



- N - Number of states

$$s_1, s_2, \cdots, s_i, \cdots s_N$$

- M - Number of symbols

$$v_1, v_2, \cdots, v_k, \cdots v_M$$

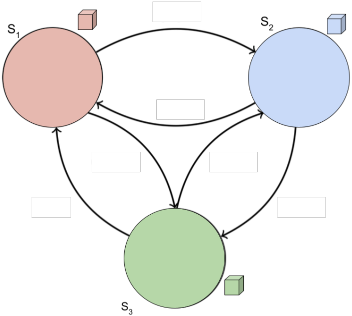
- Initial state probability

$$\Pi = [\pi_i] = P[q_1 = s_i] \qquad 1 \leq i \leq N$$

- State transition probability

$$\mathbf{A}_{N \times N} = [a_{ij}] = P[q_t = s_j / q_{t-1} = s_i]$$

HMM Parameters $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$



- N - Number of states

$$s_1, s_2, \cdots, s_i, \cdots s_N$$

- M - Number of symbols

$$v_1, v_2, \cdots, v_k, \cdots v_M$$

- Initial state probability

$$\Pi = [\pi_i] = P[q_1 = s_i] \qquad 1 \leq i \leq N$$

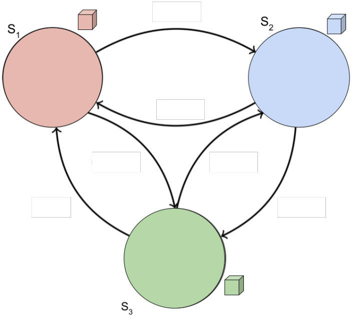
- State transition probability

$$\mathbf{A}_{N\times N} = [a_{ij}] = P[q_t = s_j / q_{t-1} = s_i]$$

- Emission Probability

$$\mathbf{B}_{N\times M} = [b_j(v_k)] = P[o_t = v_k / q_t = s_j]$$

HMM Parameters $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$



- N - Number of states

$$s_1, s_2, \cdots, s_i, \cdots s_N$$

- M - Number of symbols

$$v_1, v_2, \cdots, v_k, \cdots v_M$$

- Initial state probability

$$\Pi = [\pi_i] = P[q_1 = s_i] \qquad 1 \leq i \leq N$$

- State transition probability

$$\mathbf{A}_{N \times N} = [a_{ij}] = P[q_t = s_j / q_{t-1} = s_i]$$

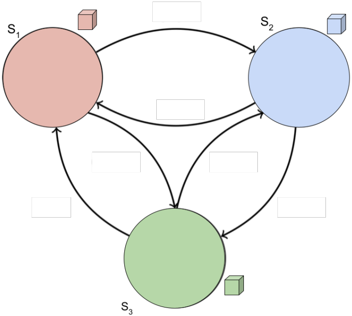
- Emission Probability

$$\mathbf{B}_{N \times M} = [b_j(v_k)] = P[o_t = v_k / q_t = s_j]$$

- Observation symbol sequence

$$O = (o_1, o_2, \cdots, o_t, \cdots o_T)$$

HMM Parameters $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$



- N - Number of states

$$s_1, s_2, \cdots, s_i, \cdots s_N$$

- M - Number of symbols

$$v_1, v_2, \cdots, v_k, \cdots v_M$$

- Initial state probability

$$\Pi = [\pi_i] = P[q_1 = s_i] \qquad 1 \leq i \leq N$$

- State transition probability

$$\mathbf{A}_{N \times N} = [a_{ij}] = P[q_t = s_j / q_{t-1} = s_i]$$

- Emission Probability

$$\mathbf{B}_{N \times M} = [b_j(v_k)] = P[o_t = v_k / q_t = s_j]$$

- Observation symbol sequence

$$O = (o_1, o_2, \cdots, o_t, \cdots o_T)$$

- Sate sequence

$$Q = (q_1, q_2, \cdots, q_t, \cdots q_T)$$

Evaluating $P[O/\lambda]$

Evaluating $P[O/\lambda]$

- Probability of producing OSS O while passing through specific SS Q

Evaluating $P[O/\lambda]$

- Probability of producing OSS O while passing through specific SS Q

$$P[O, Q/\lambda] = P[o_1, o_2, \cdots o_T, q_1, q_2, \cdots q_T/\lambda]$$

Evaluating $P[O/\lambda]$

- Probability of producing OSS O while passing through specific SS Q

$$
\begin{aligned}
P[O, Q/\lambda] &= P[o_1, o_2, \cdots o_T, q_1, q_2, \cdots q_T/\lambda] \\
&= \pi_{q_1}
\end{aligned}
$$

Evaluating $P[O/\lambda]$

- Probability of producing OSS O while passing through specific SS Q

$$
\begin{aligned}
P[O, Q/\lambda] &= P[o_1, o_2, \cdots o_T, q_1, q_2, \cdots q_T/\lambda] \\
&= \pi_{q_1} \; b_{q_1}(o_1)
\end{aligned}
$$

Evaluating $P[O/\lambda]$

- Probability of producing OSS O while passing through specific SS Q

$$
\begin{aligned}
P[O, Q/\lambda] &= P[o_1, o_2, \cdots o_T, q_1, q_2, \cdots q_T/\lambda] \\
&= \pi_{q_1}\ b_{q_1}(o_1)\ a_{q_1 q_2}\ b_{q_2}(o_2)
\end{aligned}
$$

Evaluating $P[O/\lambda]$

- Probability of producing OSS O while passing through specific SS Q

$$
\begin{aligned}
P[O, Q/\lambda] &= P[o_1, o_2, \cdots o_T, q_1, q_2, \cdots q_T/\lambda] \\
&= \pi_{q_1}\ b_{q_1}(o_1)\ a_{q_1 q_2}\ b_{q_2}(o_2) \cdots\ a_{q_{T-1} q_T}\ b_{q_T}(o_T)
\end{aligned}
$$

Evaluating $P[O/\lambda]$

- Probability of producing OSS O while passing through specific SS Q

$$
\begin{aligned}
P[O, Q/\lambda] &= P[o_1, o_2, \cdots o_T, q_1, q_2, \cdots q_T/\lambda] \\
&= \pi_{q_1}\ b_{q_1}(o_1)\ a_{q_1 q_2}\ b_{q_2}(o_2) \cdots\ a_{q_{T-1} q_T}\ b_{q_T}(o_T)
\end{aligned}
$$

- Evaluating $P[O, Q/\lambda]$ requires $2T - 1$ multiplications

Evaluating $P[O/\lambda]$

- Probability of producing OSS O while passing through specific SS Q

$$\begin{aligned} P[O, Q/\lambda] &= P[o_1, o_2, \cdots o_T, q_1, q_2, \cdots q_T/\lambda] \\ &= \pi_{q_1}\ b_{q_1}(o_1)\ a_{q_1 q_2}\ b_{q_2}(o_2) \cdots\ a_{q_{T-1} q_T}\ b_{q_T}(o_T) \end{aligned}$$

- Evaluating $P[O, Q/\lambda]$ requires $2T - 1$ multiplications
- Probability of generating OSS O given the model λ

$$P[O/\lambda] = \sum_{\text{all}Q} P[O, Q/\lambda]$$

Evaluating $P[O/\lambda]$

- Probability of producing OSS O while passing through specific SS Q

$$\begin{aligned} P[O, Q/\lambda] &= P[o_1, o_2, \cdots o_T, q_1, q_2, \cdots q_T/\lambda] \\ &= \pi_{q_1} \; b_{q_1}(o_1) \; a_{q_1 q_2} \; b_{q_2}(o_2) \cdots \; a_{q_{T-1} q_T} \; b_{q_T}(o_T) \end{aligned}$$

- Evaluating $P[O, Q/\lambda]$ requires $2T - 1$ multiplications
- Probability of generating OSS O given the model λ

$$P[O/\lambda] = \sum_{\text{all} Q} P[O, Q/\lambda]$$

- Sum over all possible N^T state sequences Q

Evaluating $P[O/\lambda]$

- Probability of producing OSS O while passing through specific SS Q

$$
\begin{aligned}
P[O, Q/\lambda] &= P[o_1, o_2, \cdots o_T, q_1, q_2, \cdots q_T/\lambda] \\
&= \pi_{q_1}\ b_{q_1}(o_1)\ a_{q_1 q_2}\ b_{q_2}(o_2) \cdots\ a_{q_{T-1} q_T}\ b_{q_T}(o_T)
\end{aligned}
$$

- Evaluating $P[O, Q/\lambda]$ requires $2T - 1$ multiplications
- Probability of generating OSS O given the model λ

$$
P[O/\lambda] = \sum_{\text{all} Q} P[O, Q/\lambda]
$$

- Sum over all possible N^T state sequences Q
- Evaluating $P[O/\lambda]$ requires $O(TN^T)$ operations.

Evaluating $P[O/\lambda]$

- Probability of producing OSS O while passing through specific SS Q

$$\begin{aligned} P[O, Q/\lambda] &= P[o_1, o_2, \cdots o_T, q_1, q_2, \cdots q_T/\lambda] \\ &= \pi_{q_1}\ b_{q_1}(o_1)\ a_{q_1 q_2}\ b_{q_2}(o_2) \cdots\ a_{q_{T-1} q_T}\ b_{q_T}(o_T) \end{aligned}$$

- Evaluating $P[O, Q/\lambda]$ requires $2T - 1$ multiplications
- Probability of generating OSS O given the model λ

$$P[O/\lambda] = \sum_{\text{all} Q} P[O, Q/\lambda]$$

- Sum over all possible N^T state sequences Q
- Evaluating $P[O/\lambda]$ requires $O(TN^T)$ operations.
- A 5-state model with 100 observations requires around 10^{72} FPOs!

HMM - Three Main Questions

- Given an observation sequence O and the model $\lambda = [\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}]$, how to efficiently evaluate $P(O/\lambda)$?

HMM - Three Main Questions

- Given an observation sequence O and the model $\lambda = [\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}]$, how to efficiently evaluate $P(O/\lambda)$?
  - Brute-force evaluation requires a complexity of $O(TN^T)$.
  - Forward backward algorithm can be used to evaluate in $O(N^2T)$

HMM - Three Main Questions

- Given an observation sequence O and the model $\lambda = [\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}]$, how to efficiently evaluate $P(O/\lambda)$?
  - Brute-force evaluation requires a complexity of $O(TN^T)$.
  - Forward backward algorithm can be used to evaluate in $O(N^2T)$
- Given an observation sequence O and the model $\lambda = [\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}]$, how to determine the optimal state sequence?

HMM - Three Main Questions

- Given an observation sequence O and the model $\lambda = [\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}]$, how to efficiently evaluate $P(O/\lambda)$?
  - Brute-force evaluation requires a complexity of $O(TN^T)$.
  - Forward backward algorithm can be used to evaluate in $O(N^2T)$
- Given an observation sequence O and the model $\lambda = [\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}]$, how to determine the optimal state sequence?
  - Brute-force evaluation requires a comparison across N^T paths
  - Viterbi algorithm can be used to determine the best path in $O(N^2T)$

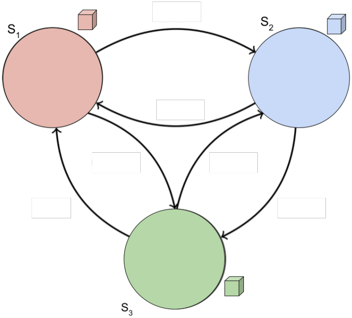
HMM - Three Main Questions

- Given an observation sequence O and the model $\lambda = [\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}]$, how to efficiently evaluate $P(O/\lambda)$?
  - Brute-force evaluation requires a complexity of $O(TN^T)$.
  - Forward backward algorithm can be used to evaluate in $O(N^2T)$
- Given an observation sequence O and the model $\lambda = [\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}]$, how to determine the optimal state sequence?
  - Brute-force evaluation requires a comparison across N^T paths
  - Viterbi algorithm can be used to determine the best path in $O(N^2T)$
- Given the observed data O, how to evaluate the model parameters?

HMM - Three Main Questions

- Given an observation sequence O and the model $\lambda = [\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}]$, how to efficiently evaluate $P(O/\lambda)$?
  - Brute-force evaluation requires a complexity of $O(TN^T)$.
  - Forward backward algorithm can be used to evaluate in $O(N^2T)$
- Given an observation sequence O and the model $\lambda = [\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}]$, how to determine the optimal state sequence?
  - Brute-force evaluation requires a comparison across N^T paths
  - Viterbi algorithm can be used to determine the best path in $O(N^2T)$
- Given the observed data O, how to evaluate the model parameters?
  - Maximizing likelihood through gradient-ascent would be very slow.
  - Baum-Welch algorithm can be used to estimate the model parameters.

HMM Parameters $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$ (Recap)



- N - Number of states

$$s_1, s_2, \cdots, s_i, \cdots s_N$$

- M - Number of symbols

$$v_1, v_2, \cdots, v_k, \cdots v_M$$

- Initial state probability

$$\Pi = [\pi_i] = P[q_1 = s_i] \qquad 1 \leq i \leq N$$

- State transition probability

$$\mathbf{A}_{N\times N} = [a_{ij}] = P[q_t = s_j / q_{t-1} = s_i]$$

- Emission Probability

$$\mathbf{B}_{N\times M} = [b_j(v_k)] = P[o_t = v_k / q_t = s_j]$$

- Observation symbol sequence

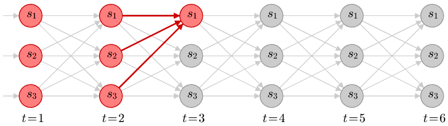
$$O = (o_1, o_2, \cdots, o_t, \cdots o_T)$$

- Sate sequence

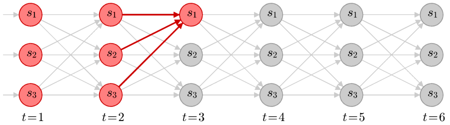
$$Q = (q_1, q_2, \cdots, q_t, \cdots q_T)$$

Forward Variable $\alpha_t(i)$

Forward Variable $\alpha_t(i)$

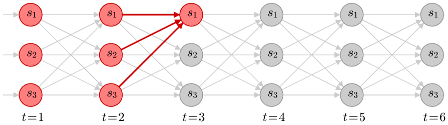


Forward Variable $\alpha_t(i)$



- Probability of producing a partial OSS $(o_1, o_2 \cdots o_t)$ ending in state s_i at time (t) given the model λ

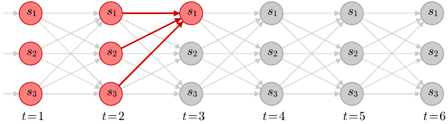
Forward Variable $\alpha_t(i)$



- Probability of producing a partial OSS $(o_1, o_2 \cdots o_t)$ ending in state s_i at time (t) given the model λ

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

Forward Variable $\alpha_t(i)$



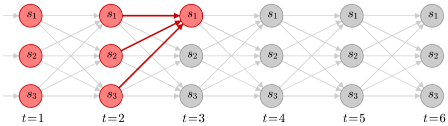
- Initialization (at t=1)

$$\alpha_1(i) =$$

- Probability of producing a partial OSS $(o_1, o_2 \cdots o_t)$ ending in state s_i at time (t) given the model λ

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

Forward Variable $\alpha_t(i)$



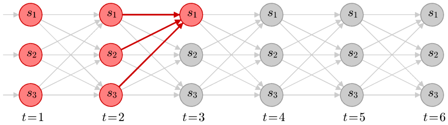
- Initialization (at t=1)

$$\alpha_1(i) = \pi_i b_i(o_1) \quad 1 \leq i \leq N$$

- Probability of producing a partial OSS $(o_1, o_2 \cdots o_t)$ ending in state s_i at time (t) given the model λ

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

Forward Variable $\alpha_t(i)$



- Probability of producing a partial OSS $(o_1, o_2 \cdots o_t)$ ending in state s_i at time (t) given the model λ

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

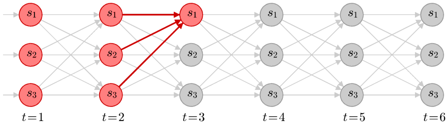
- Initialization (at t=1)

$$\alpha_1(i) = \pi_i b_i(o_1) \quad 1 \leq i \leq N$$

- Induction (given $\alpha_t(i)$)

$$\alpha_{t+1}(j) =$$

Forward Variable $\alpha_t(i)$



- Probability of producing a partial OSS $(o_1, o_2 \cdots o_t)$ ending in state s_i at time (t) given the model λ

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

- Initialization (at t=1)

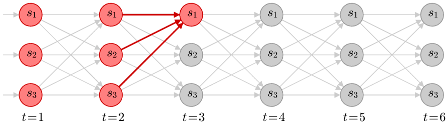
$$\alpha_1(i) = \pi_i b_i(o_1) \quad 1 \leq i \leq N$$

- Induction (given $\alpha_t(i)$)

$$\alpha_{t+1}(j) = \sum_{i=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1})$$

$$1 \leq i \leq N, \quad 1 \leq t \leq T-1$$

Forward Variable $\alpha_t(i)$



- Probability of producing a partial OSS $(o_1, o_2 \cdots o_t)$ ending in state s_i at time (t) given the model λ

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

- Initialization (at t=1)

$$\alpha_1(i) = \pi_i b_i(o_1) \quad 1 \leq i \leq N$$

- Induction (given $\alpha_t(i)$)

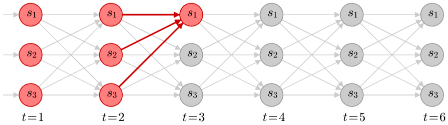
$$\alpha_{t+1}(j) = \sum_{i=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1})$$

$$1 \leq i \leq N, \quad 1 \leq t \leq T-1$$

- Termination (given $\alpha_T(i)$)

$$P[O/\lambda] =$$

Forward Variable $\alpha_t(i)$



- Probability of producing a partial OSS $(o_1, o_2 \cdots o_t)$ ending in state s_i at time (t) given the model λ

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

- Initialization (at t=1)

$$\alpha_1(i) = \pi_i b_i(o_1) \quad 1 \leq i \leq N$$

- Induction (given $\alpha_t(i)$)

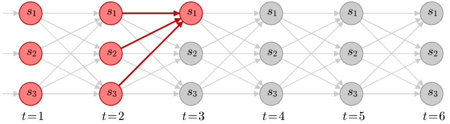
$$\alpha_{t+1}(j) = \sum_{i=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1})$$

$$1 \leq i \leq N, \quad 1 \leq t \leq T-1$$

- Termination (given $\alpha_T(i)$)

$$P[O/\lambda] = \sum_{i=1}^{N} \alpha_T(i)$$

Forward Variable $\alpha_t(i)$



- Probability of producing a partial OSS $(o_1, o_2 \cdots o_t)$ ending in state s_i at time (t) given the model λ

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

- Initialization (at t=1)

$$\alpha_1(i) = \pi_i b_i(o_1) \quad 1 \leq i \leq N$$

- Induction (given $\alpha_t(i)$)

$$\alpha_{t+1}(j) = \sum_{i=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1})$$

$$1 \leq i \leq N, \quad 1 \leq t \leq T-1$$

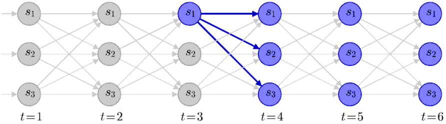
- Termination (given $\alpha_T(i)$)

$$P[O/\lambda] = \sum_{i=1}^{N} \alpha_T(i)$$

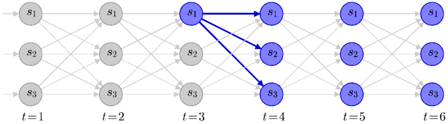
- Complexity: $O(N^2T)$

Backward Variable $\beta_t(i)$

Backward Variable $\beta_t(i)$

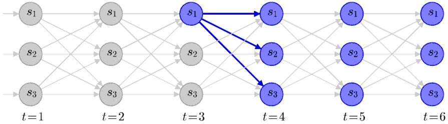


Backward Variable $\beta_t(i)$



- Probability of producing a partial OSS $(o_{t+1}, o_{t+2} \cdots o_T)$ given state s_i at time (t) and model λ

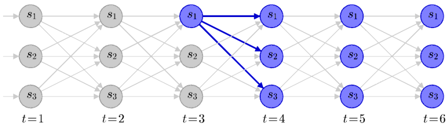
Backward Variable $\beta_t(i)$



- Probability of producing a partial OSS $(o_{t+1}, o_{t+2} \cdots o_T)$ given state s_i at time (t) and model λ

$$\beta_t(i) = P[o_{t+1}, o_{t+2} \cdots o_T / q_t = s_i, \lambda]$$

Backward Variable $\beta_t(i)$



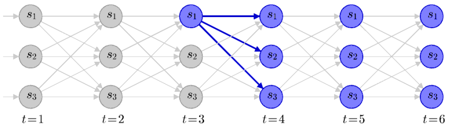
- Initialization (at t=T)

$$\beta_T(i) =$$

- Probability of producing a partial OSS $(o_{t+1}, o_{t+2} \cdots o_T)$ given state s_i at time (t) and model λ

$$\beta_t(i) = P[o_{t+1}, o_{t+2} \cdots o_T / q_t = s_i, \lambda]$$

Backward Variable $\beta_t(i)$



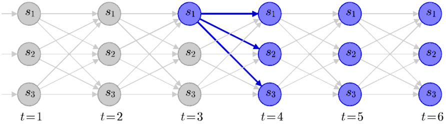
- Initialization (at t=T)

$$\beta_T(i) = 1 \quad 1 \leq i \leq N$$

- Probability of producing a partial OSS $(o_{t+1}, o_{t+2} \cdots o_T)$ given state s_i at time (t) and model λ

$$\beta_t(i) = P[o_{t+1}, o_{t+2} \cdots o_T / q_t = s_i, \lambda]$$

Backward Variable $\beta_t(i)$



- Initialization (at t=T)

$$\beta_T(i) = 1 \quad 1 \leq i \leq N$$

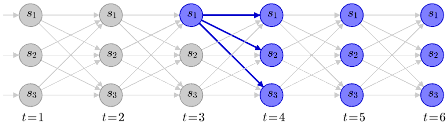
- Induction (given $\beta_{t+1}(j)$)

$$\beta_t(i) =$$

- Probability of producing a partial OSS $(o_{t+1}, o_{t+2} \cdots o_T)$ given state s_i at time (t) and model λ

$$\beta_t(i) = P[o_{t+1}, o_{t+2} \cdots o_T / q_t = s_i, \lambda]$$

Backward Variable $\beta_t(i)$



- Probability of producing a partial OSS $(o_{t+1}, o_{t+2} \cdots o_T)$ given state s_i at time (t) and model λ

$$\beta_t(i) = P[o_{t+1}, o_{t+2} \cdots o_T / q_t = s_i, \lambda]$$

- Initialization (at t=T)

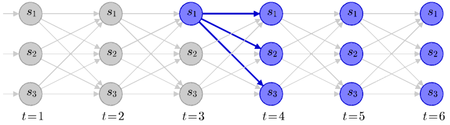
$$\beta_T(i) = 1 \quad 1 \leq i \leq N$$

- Induction (given $\beta_{t+1}(j)$)

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$$

$$1 \leq i \leq N, \quad t = T-1, \cdots 1$$

Backward Variable $\beta_t(i)$



- Probability of producing a partial OSS $(o_{t+1}, o_{t+2} \cdots o_T)$ given state s_i at time (t) and model λ

$$\beta_t(i) = P[o_{t+1}, o_{t+2} \cdots o_T / q_t = s_i, \lambda]$$

- Initialization (at t=T)

$$\beta_T(i) = 1 \quad 1 \le i \le N$$

- Induction (given $\beta_{t+1}(j)$)

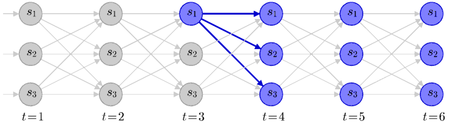
$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$$

$$1 \le i \le N, \quad t = T-1, \cdots 1$$

- Termination (given $\beta_1(i)$)

$$P[O/\lambda] =$$

Backward Variable $\beta_t(i)$



- Probability of producing a partial OSS $(o_{t+1}, o_{t+2} \cdots o_T)$ given state s_i at time (t) and model λ

$$\beta_t(i) = P[o_{t+1}, o_{t+2} \cdots o_T / q_t = s_i, \lambda]$$

- Initialization (at t=T)

$$\beta_T(i) = 1 \quad 1 \leq i \leq N$$

- Induction (given $\beta_{t+1}(j)$)

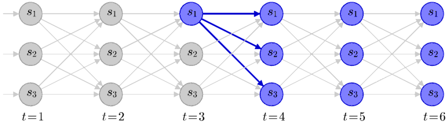
$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$$

$$1 \leq i \leq N, \quad t = T-1, \cdots 1$$

- Termination (given $\beta_1(i)$)

$$P[O/\lambda] = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i)$$

Backward Variable $\beta_t(i)$



- Probability of producing a partial OSS $(o_{t+1}, o_{t+2} \cdots o_T)$ given state s_i at time (t) and model λ

$$\beta_t(i) = P[o_{t+1}, o_{t+2} \cdots o_T / q_t = s_i, \lambda]$$

- Initialization (at t=T)

$$\beta_T(i) = 1 \quad 1 \leq i \leq N$$

- Induction (given $\beta_{t+1}(j)$)

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$$

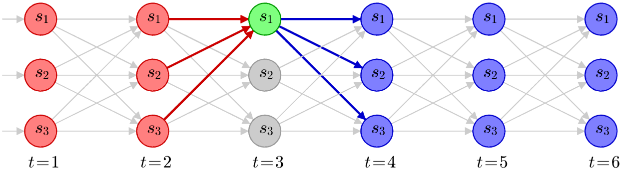
$$1 \leq i \leq N, \quad t = T-1, \cdots 1$$

- Termination (given $\beta_1(i)$)

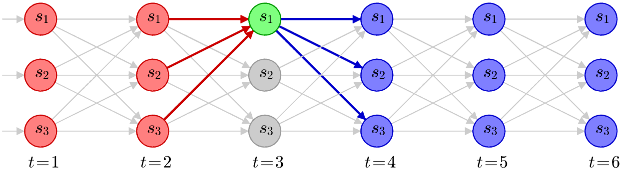
$$P[O/\lambda] = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i)$$

- Complexity: $O(N^2 T)$

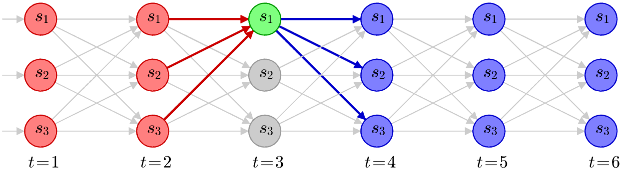
Probability of State



Probability of State



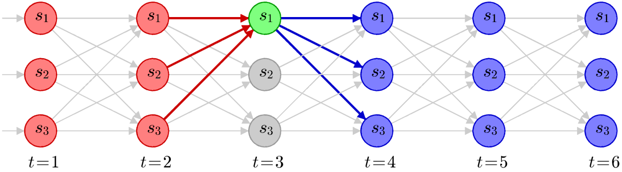
Probability of State



- Probability of being in state s_i at time t, given OSS O and model λ

$$\gamma_t(i) \quad = \quad P[q_t = s_i / O, \lambda]$$

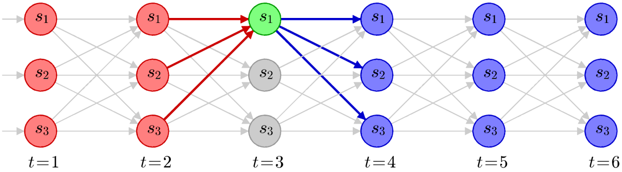
Probability of State



- Probability of being in state s_i at time t, given OSS O and model λ

$$\begin{aligned}\gamma_t(i) &= P[q_t = s_i/O, \lambda] \\ &= \frac{P[q_t = s_i, O/\lambda]}{P[O/\lambda]}\end{aligned}$$

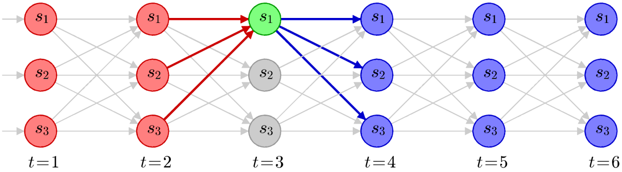
Probability of State



- Probability of being in state s_i at time t, given OSS O and model λ

$$\begin{aligned}\gamma_t(i) &= P[q_t = s_i/O, \lambda] \\ &= \frac{P[q_t = s_i, O/\lambda]}{P[O/\lambda]} \\ &= \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}\end{aligned}$$

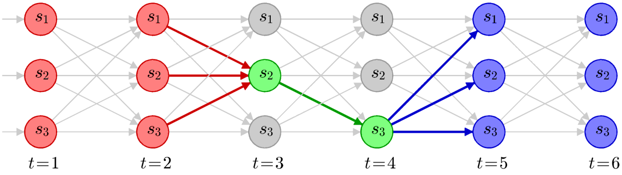
Probability of State



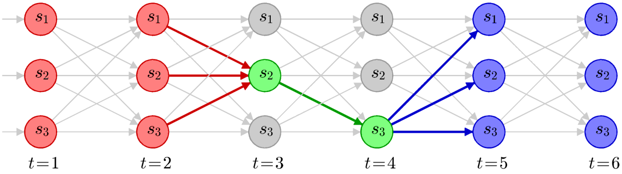
- Probability of being in state s_i at time t, given OSS O and model λ

$$\begin{aligned}\gamma_t(i) &= P[q_t = s_i / O, \lambda] \\ &= \frac{P[q_t = s_i, O/\lambda]}{P[O/\lambda]} \\ &= \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}\end{aligned}$$

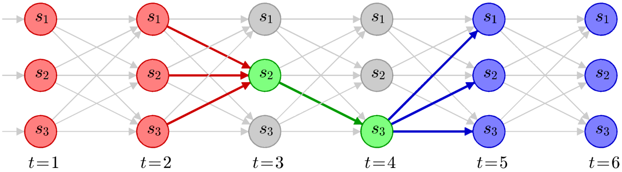
Probability of State Transition



Probability of State Transition



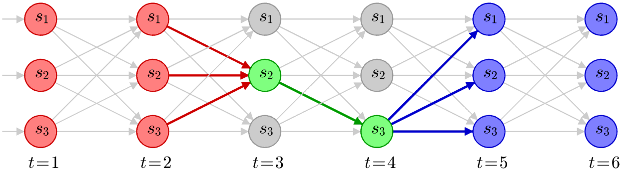
Probability of State Transition



- Probability of being in state s_i at time t and state s_j at time $t+1$, given OSS O and model λ

$$\xi_t(i,j) \quad = \quad P[q_t = s_i, q_{t+1} = s_j / O, \lambda]$$

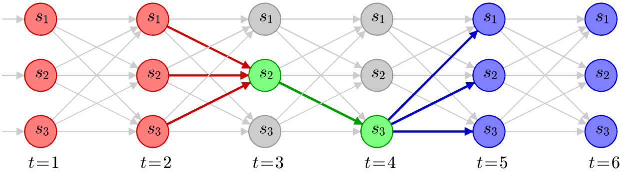
Probability of State Transition



- Probability of being in state s_i at time t and state s_j at time $t+1$, given OSS O and model λ

$$\begin{aligned}\xi_t(i,j) &= P[q_t = s_i, q_{t+1} = s_j / O, \lambda] \\ &= \frac{P[q_t = s_i, q_{t+1} = s_j, O/\lambda]}{P[O/\lambda]}\end{aligned}$$

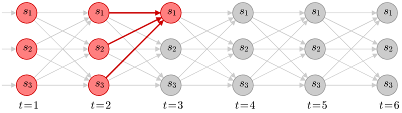
Probability of State Transition



- Probability of being in state s_i at time t and state s_j at time $t+1$, given OSS O and model λ

$$
\begin{aligned}
\xi_t(i,j) &= P[q_t = s_i, q_{t+1} = s_j / O, \lambda] \\
&= \frac{P[q_t = s_i, q_{t+1} = s_j, O/\lambda]}{P[O/\lambda]} \\
&= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}
\end{aligned}
$$

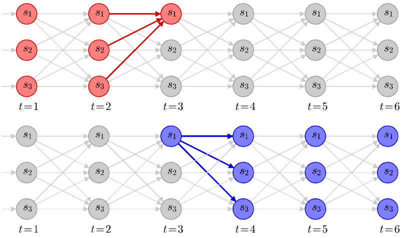
HMM Variables (Recap)



Forward variable:

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

HMM Variables (Recap)



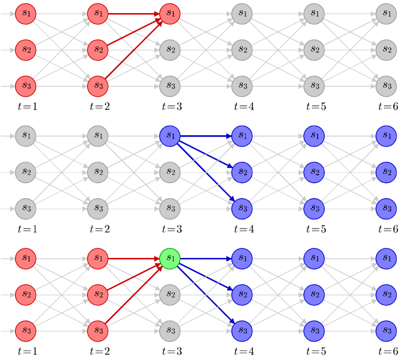
Forward variable:

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

Backward variable:

$$\beta_t(i) = P[o_{t+1}, o_{t+2} \cdots o_T/q_t = s_i, \lambda]$$

HMM Variables (Recap)



Forward variable:

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

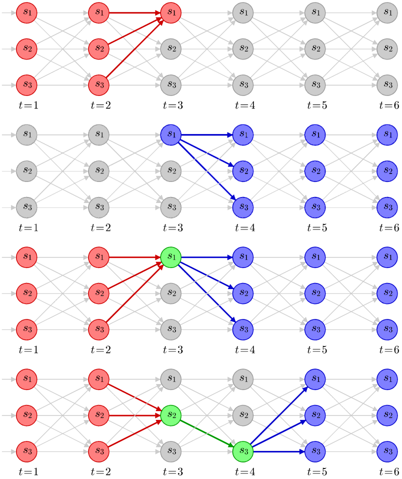
Backward variable:

$$\beta_t(i) = P[o_{t+1}, o_{t+2} \cdots o_T/q_t = s_i, \lambda]$$

State marginals:

$$\gamma_t(i) = P[q_t = s_i/O, \lambda]$$

HMM Variables (Recap)



Forward variable:

$$\alpha_t(i) = P[o_1, o_2, \cdots o_t, q_t = s_i/\lambda]$$

Backward variable:

$$\beta_t(i) = P[o_{t+1}, o_{t+2} \cdots o_T/q_t = s_i, \lambda]$$

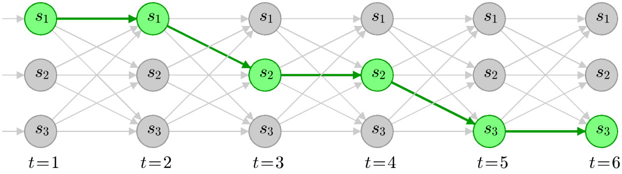
State marginals:

$$\gamma_t(i) = P[q_t = s_i/O, \lambda]$$

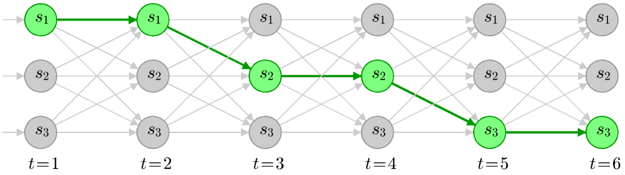
State-transition marginals:

$$\xi_t(i,j) = P[q_t = s_i, q_{t+1} = s_j/O, \lambda]$$

Optimal State Sequence



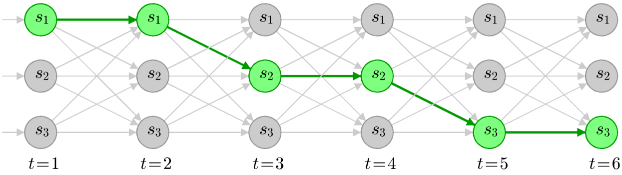
Optimal State Sequence



- Can local-best states result in global-best path?

$$q_t = \arg\max_{1 \le i \le N} \gamma_t(i) \qquad 1 \le t \le T$$

Optimal State Sequence



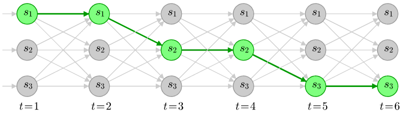
- Can local-best states result in global-best path?

$$q_t = \arg\max_{1 \leq i \leq N} \gamma_t(i) \qquad 1 \leq t \leq T$$

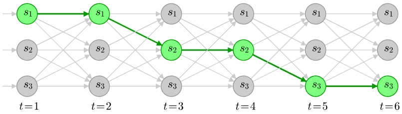
- Probability along the single best path ending in state S_i while generating the partial OSS $(o_1, o_2, \ldots o_t)$ given the

$$\delta_t(i) = \max_{q_1, q_2, \ldots q_{t-1}} P[o_1, o_2, \cdots o_t, q_1, q_2, \cdots q_t = s_i / \lambda]$$

Viterbi Algorithm

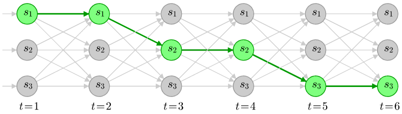


Viterbi Algorithm



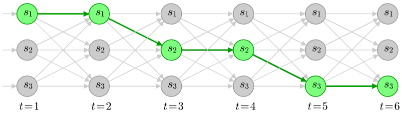
- Probability of producing a P-OSS $o_1, \cdots o_t$ along single best path ending in $q_t = S_i$.

Viterbi Algorithm



- Probability of producing a P-OSS $o_1, \cdots o_t$ along single best path ending in $q_t = S_i$.
- Initialization

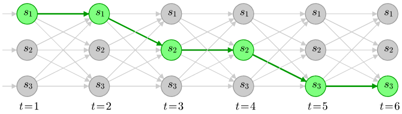
Viterbi Algorithm



- Probability of producing a P-OSS $o_1, \cdots o_t$ along single best path ending in $q_t = S_i$.
- Initialization

$$\begin{aligned} \delta_1(i) &= \pi_i b_i(o_1) \\ \psi_1(i) &= 0 \quad 1 \leq i \leq N \end{aligned}$$

Viterbi Algorithm

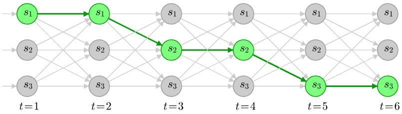


- Probability of producing a P-OSS $o_1, \cdots o_t$ along single best path ending in $q_t = S_i$.
- Initialization

$$\begin{aligned} \delta_1(i) &= \pi_i b_i(o_1) \\ \psi_1(i) &= 0 \quad 1 \leq i \leq N \end{aligned}$$

- Induction (given $\delta_{t-1}(i)$)

Viterbi Algorithm



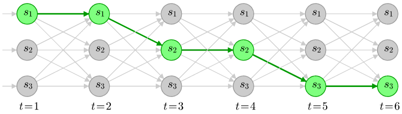
- Probability of producing a P-OSS $o_1, \cdots o_t$ along single best path ending in $q_t = S_i$.
- Initialization

$$\delta_1(i) = \pi_i b_i(o_1)$$
$$\psi_1(i) = 0 \quad 1 \leq i \leq N$$

- Induction (given $\delta_{t-1}(i)$)

$$\delta_t(j) = \max_{1 \leq i \leq N} \delta_{t-1}(i) a_{ij} b_j(o_t)$$
$$\psi_t(j) = \arg \max_{1 \leq i \leq N} \delta_{t-1}(i) a_{ij}$$
$$1 \leq j \leq N \quad 1 < t \leq T$$

Viterbi Algorithm



- Probability of producing a P-OSS $o_1, \cdots o_t$ along single best path ending in $q_t = S_i$.
- Initialization

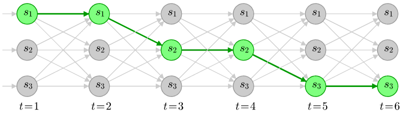
$$\delta_1(i) = \pi_i b_i(o_1)$$
$$\psi_1(i) = 0 \quad 1 \leq i \leq N$$

- Induction (given $\delta_{t-1}(i)$)

$$\delta_t(j) = \max_{1 \leq i \leq N} \delta_{t-1}(i) a_{ij} b_j(o_t)$$
$$\psi_t(j) = \arg \max_{1 \leq i \leq N} \delta_{t-1}(i) a_{ij}$$
$$1 \leq j \leq N \quad 1 < t \leq T$$

- Termination (given $\delta_T(i)$)

Viterbi Algorithm



- Probability of producing a P-OSS $o_1, \cdots o_t$ along single best path ending in $q_t = S_i$.
- Initialization

$$\delta_1(i) = \pi_i b_i(o_1)$$
$$\psi_1(i) = 0 \quad 1 \leq i \leq N$$

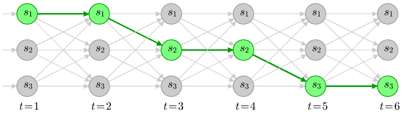
- Induction (given $\delta_{t-1}(i)$)

$$\delta_t(j) = \max_{1\leq i\leq N} \delta_{t-1}(i) a_{ij} b_j(o_t)$$
$$\psi_t(j) = \arg\max_{1\leq i\leq N} \delta_{t-1}(i) a_{ij}$$
$$1 \leq j \leq N \quad 1 < t \leq T$$

- Termination (given $\delta_T(i)$)

$$P^* = \max_{1\leq i\leq N} \delta_T(i)$$
$$q_T^* = \arg\max_{1\leq i\leq N} \delta_T(i)$$

Viterbi Algorithm



- Probability of producing a P-OSS $o_1, \cdots o_t$ along single best path ending in $q_t = S_i$.
- Initialization

$$\begin{aligned} \delta_1(i) &= \pi_i b_i(o_1) \\ \psi_1(i) &= 0 \quad 1 \le i \le N \end{aligned}$$

- Induction (given $\delta_{t-1}(i)$)

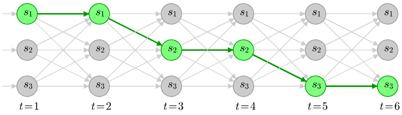
$$\begin{aligned} \delta_t(j) &= \max_{1 \le i \le N} \delta_{t-1}(i) a_{ij} b_j(o_t) \\ \psi_t(j) &= \arg \max_{1 \le i \le N} \delta_{t-1}(i) a_{ij} \\ & \quad 1 \le j \le N \quad 1 < t \le T \end{aligned}$$

- Termination (given $\delta_T(i)$)

$$\begin{aligned} P^* &= \max_{1 \le i \le N} \delta_T(i) \\ q_T^* &= \arg \max_{1 \le i \le N} \delta_T(i) \end{aligned}$$

- Back-tracking

Viterbi Algorithm



- Probability of producing a P-OSS $o_1, \cdots o_t$ along single best path ending in $q_t = S_i$.
- Initialization

$$\delta_1(i) = \pi_i b_i(o_1)$$
$$\psi_1(i) = 0 \quad 1 \leq i \leq N$$

- Induction (given $\delta_{t-1}(i)$)

$$\delta_t(j) = \max_{1 \leq i \leq N} \delta_{t-1}(i) a_{ij} b_j(o_t)$$
$$\psi_t(j) = \arg\max_{1 \leq i \leq N} \delta_{t-1}(i) a_{ij}$$
$$1 \leq j \leq N \quad 1 < t \leq T$$

- Termination (given $\delta_T(i)$)

$$P^* = \max_{1 \leq i \leq N} \delta_T(i)$$
$$q_T^* = \arg\max_{1 \leq i \leq N} \delta_T(i)$$

- Back-tracking

$$q_t^* = \psi_{t+1}(q_{t+1}^*)$$

$$t = T-1, T-2, \cdots 1$$

HMM Parameter Estimation

HMM Parameter Estimation

- Given the OSS O, estimate the model parameters λ

HMM Parameter Estimation

- Given the OSS O, estimate the model parameters λ
- Estimate λ to maximize the log-likelihood

$$\mathcal{L}(\lambda) = \log P[O/\lambda]$$

HMM Parameter Estimation

- Given the OSS O, estimate the model parameters λ
- Estimate λ to maximize the log-likelihood

$$\mathcal{L}(\lambda) = \log P[O/\lambda]$$

- Constraints

$$\sum_{i=1}^{N} \pi_i = 1$$

HMM Parameter Estimation

- Given the OSS O, estimate the model parameters λ
- Estimate λ to maximize the log-likelihood

$$\mathcal{L}(\lambda) = \log P[O/\lambda]$$

- Constraints

$$\sum_{i=1}^{N} \pi_i = 1 \qquad \sum_{j=1}^{N} a_{ij} = 1$$

HMM Parameter Estimation

- Given the OSS O, estimate the model parameters λ
- Estimate λ to maximize the log-likelihood

$$\mathcal{L}(\lambda) = \log P[O/\lambda]$$

- Constraints

$$\sum_{i=1}^{N} \pi_i = 1 \qquad \sum_{j=1}^{N} a_{ij} = 1 \qquad \sum_{k=1}^{M} b_j(v_k) = 1$$

HMM Parameter Estimation

- Given the OSS O, estimate the model parameters λ
- Estimate λ to maximize the log-likelihood

$$\mathcal{L}(\lambda) = \log P[O/\lambda]$$

- Constraints

$$\sum_{i=1}^{N} \pi_i = 1 \qquad \sum_{j=1}^{N} a_{ij} = 1 \qquad \sum_{k=1}^{M} b_j(v_k) = 1$$

- Gradient ascent would be extremely slow

HMM Parameter Estimation

- Given the OSS O, estimate the model parameters λ
- Estimate λ to maximize the log-likelihood

$$\mathcal{L}(\lambda) = \log P[O/\lambda]$$

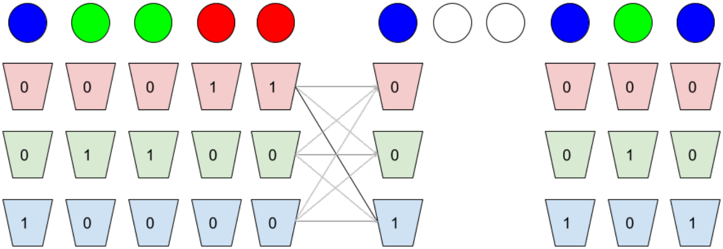
- Constraints

$$\sum_{i=1}^{N} \pi_i = 1 \qquad \sum_{j=1}^{N} a_{ij} = 1 \qquad \sum_{k=1}^{M} b_j(v_k) = 1$$

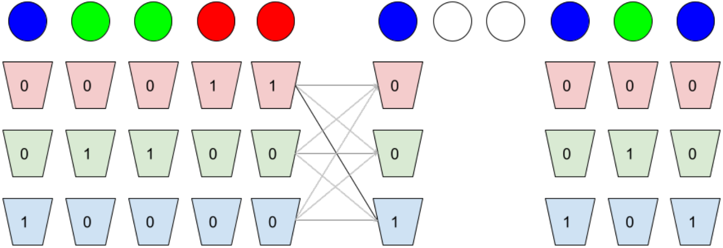
- Gradient ascent would be extremely slow
- Baum-Welch algorithm is used for iterative estimation

Parameter Estimation - Markov Model

Parameter Estimation - Markov Model

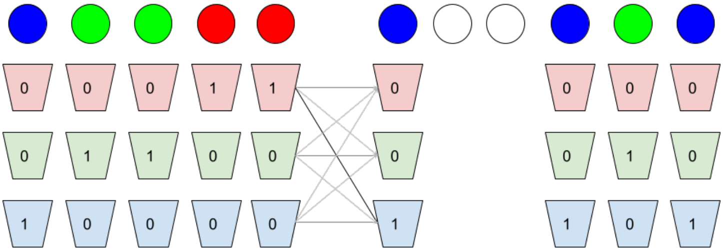


Parameter Estimation - Markov Model



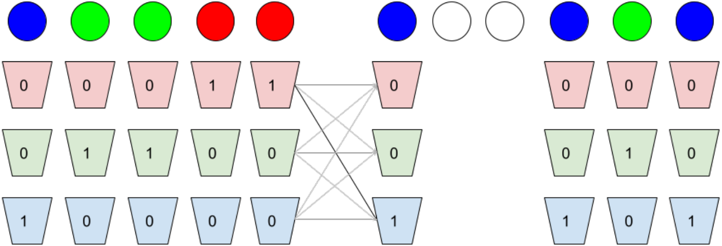
$$\hat{\pi}_R =$$

Parameter Estimation - Markov Model



$$\hat{\pi}_R = \text{Probability of starting from R} =$$

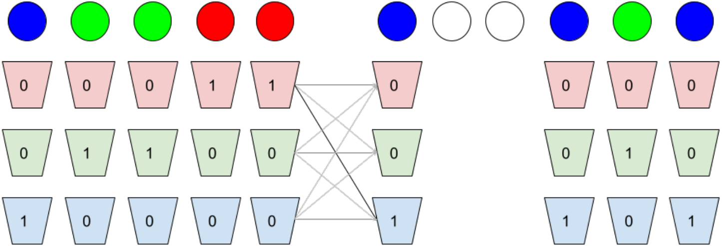
Parameter Estimation - Markov Model



$$\hat{\pi}_R = \text{Probability of starting from R} = \gamma_1(R)$$

$$\hat{a}_{RB} =$$

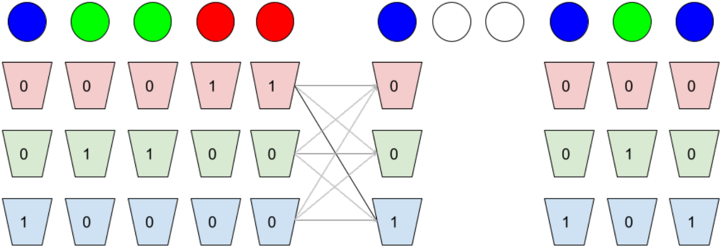
Parameter Estimation - Markov Model



$$\hat{\pi}_R \quad = \text{Probability of starting from R} \quad = \gamma_1(R)$$

$$\hat{a}_{RB} \quad = \frac{\text{Number of transitions from R to B}}{\text{Number of transitions from R}} \quad =$$

Parameter Estimation - Markov Model

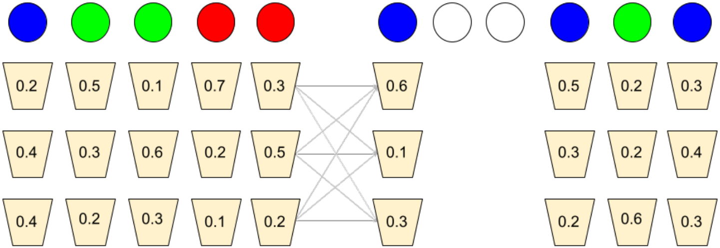


$$\hat{\pi}_R = \text{Probability of starting from R} = \gamma_1(R)$$

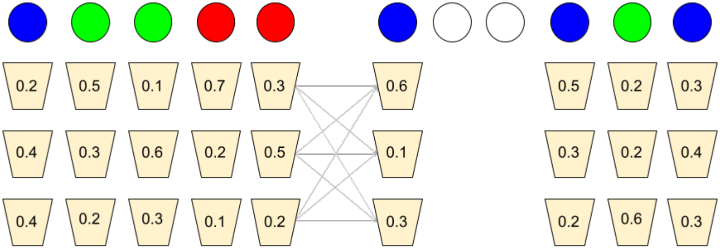
$$\hat{a}_{RB} = \frac{\text{Number of transitions from R to B}}{\text{Number of transitions from R}} = \frac{\sum_{t=1}^{T-1} \xi_t(R,B)}{\sum_{t=1}^{T-1} \gamma_t(R)}$$

Parameter Estimation - HMM

Parameter Estimation - HMM

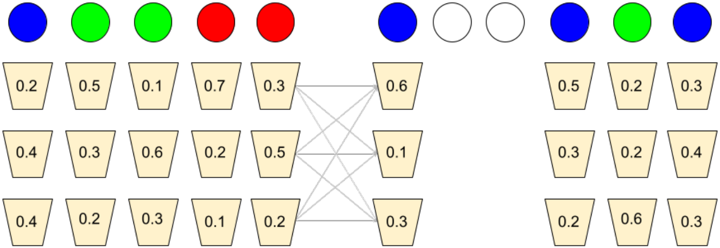


Parameter Estimation - HMM



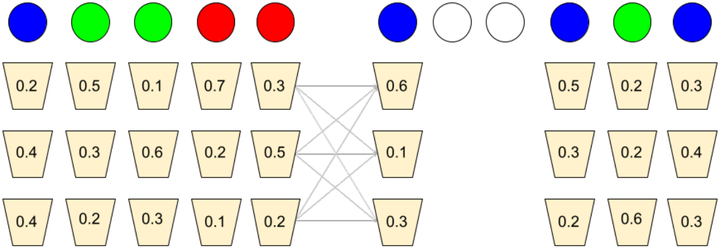
$$\hat{\pi}_i \qquad =$$

Parameter Estimation - HMM



$$\hat{\pi}_i \quad = \text{Probability of starting from } s_i \quad =$$

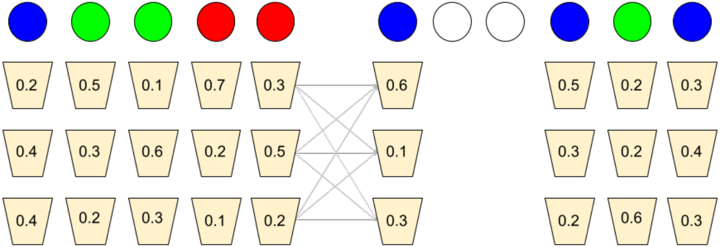
Parameter Estimation - HMM



$$\hat{\pi}_i \quad = \text{Probability of starting from } s_i \quad = \gamma_1(i)$$

$$\hat{a}_{ij} \quad =$$

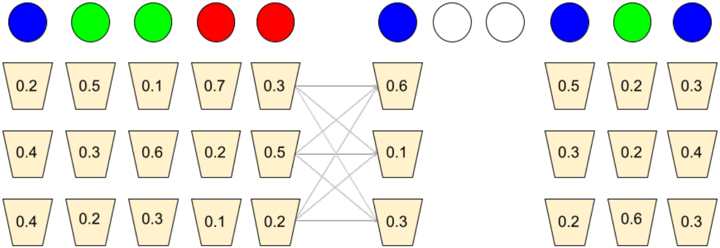
Parameter Estimation - HMM



$$\hat{\pi}_i \quad = \text{Probability of starting from } s_i \quad = \gamma_1(i)$$

$$\hat{a}_{ij} \quad = \frac{\text{Expected number of transitions from } s_i \text{ to } s_j}{\text{Expected number of transitions from } s_i} \quad =$$

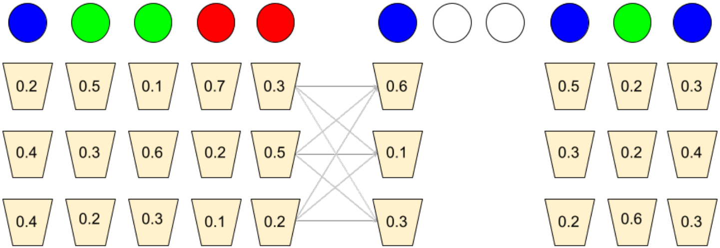
Parameter Estimation - HMM



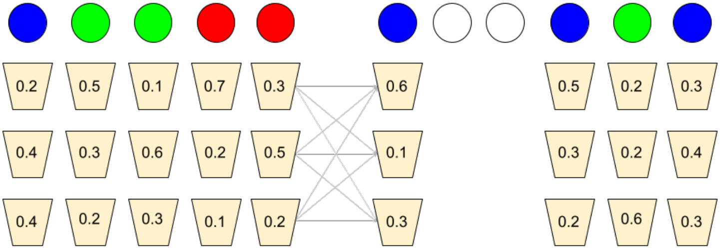
$$\hat{\pi}_i = \text{Probability of starting from } s_i = \gamma_1(i)$$

$$\hat{a}_{ij} = \frac{\text{Expected number of transitions from } s_i \text{ to } s_j}{\text{Expected number of transitions from } s_i} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

Parameter Estimation - HMM

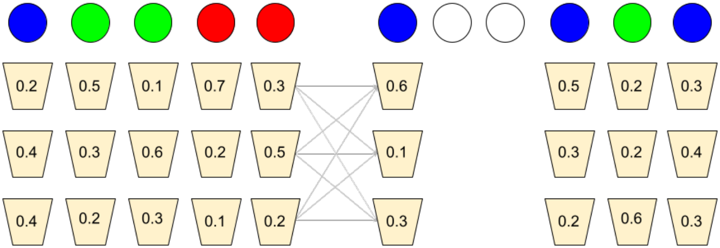


Parameter Estimation - HMM



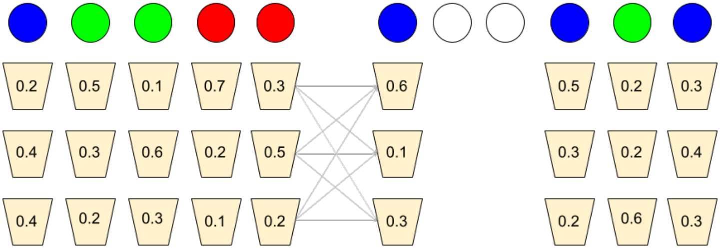
$$\hat{b}_j(v_k) \quad =$$

Parameter Estimation - HMM



$$\hat{b}_j(v_k) = \frac{\text{Expected number of times in } s_j \text{ observing } v_k}{\text{Expected number of times in } s_j} =$$

Parameter Estimation - HMM



$$\hat{b}_j(v_k) = \frac{\text{Expected number of times in } s_j \text{ observing } v_k}{\text{Expected number of times in } s_j} = \frac{\displaystyle\sum_{\substack{t=1 \\ \text{s.t } o_t = v_k}}^{T} \gamma_t(j)}{\displaystyle\sum_{t=1}^{T} \gamma_t(j)}$$

Baum-Welch Algorithm

Baum-Welch Algorithm

- Initialize HMM parameters $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$ randomly such that

$$\sum_{i=1}^{N} \pi_i = 1 \qquad \sum_{j=1}^{N} a_{ij} = 1 \qquad \sum_{k=1}^{M} b_j(v_k) = 1$$

Baum-Welch Algorithm

- Initialize HMM parameters $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$ randomly such that

$$\sum_{i=1}^{N} \pi_i = 1 \qquad \sum_{j=1}^{N} a_{ij} = 1 \qquad \sum_{k=1}^{M} b_j(v_k) = 1$$

- Compute the forward and backward variables $\alpha_t(i)$ and $\beta_t(i)$

Baum-Welch Algorithm

- Initialize HMM parameters $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$ randomly such that

$$\sum_{i=1}^{N} \pi_i = 1 \qquad \sum_{j=1}^{N} a_{ij} = 1 \qquad \sum_{k=1}^{M} b_j(v_k) = 1$$

- Compute the forward and backward variables $\alpha_t(i)$ and $\beta_t(i)$
- **Expectation step**: Evaluate Baum-Welch statistics

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \qquad \xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}$$

Baum-Welch Algorithm

- Initialize HMM parameters $\lambda = (\mathbf{A}, \mathbf{B}, \pi)$ randomly such that

$$\sum_{i=1}^{N} \pi_i = 1 \qquad \sum_{j=1}^{N} a_{ij} = 1 \qquad \sum_{k=1}^{M} b_j(v_k) = 1$$

- Compute the forward and backward variables $\alpha_t(i)$ and $\beta_t(i)$
- **Expectation step**: Evaluate Baum-Welch statistics

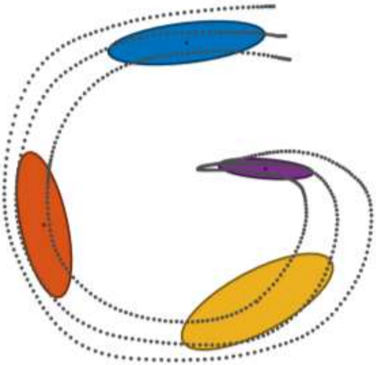
$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \qquad \xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1})\beta_{t+1}(j)}$$

- **Maximization step**: Update HMM parameters

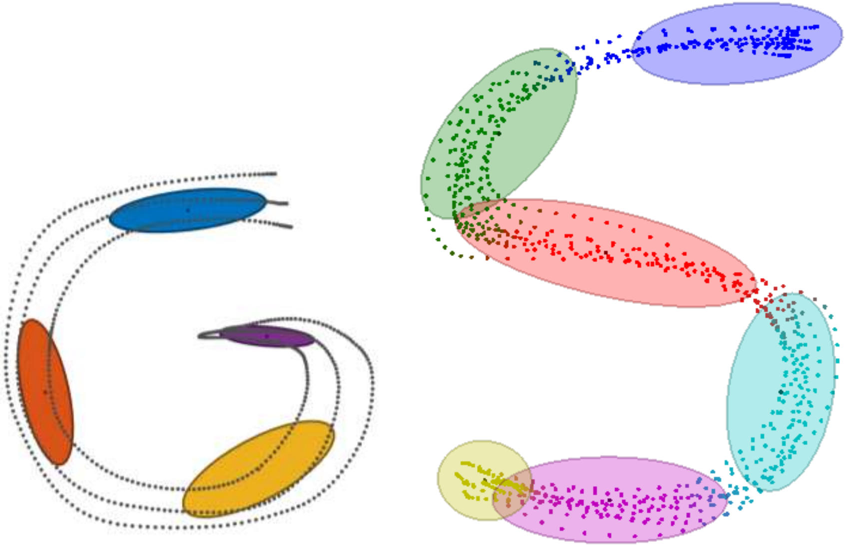
$$\hat{\pi}_i = \gamma_1(i) \qquad \hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \qquad \hat{b}_j(v_k) = \frac{\sum_{t=1,\ \text{s.t}\ o_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

Continuous Process

Continuous Process



Continuous Process



Continuous Density HMMs

Continuous Density HMMs

- CD-HMMs are used to model continuous valued observations

Continuous Density HMMs

- CD-HMMs are used to model continuous valued observations
- State is modeled as a continuous random variable

Continuous Density HMMs

- CD-HMMs are used to model continuous valued observations
- State is modeled as a continuous random variable
- Eg. Emission probability of CD-HMMs can be modeled using Gaussian

$$b_j(\mathbf{v}) \sim \mathcal{N}(\mathbf{v}/\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

Continuous Density HMMs

- CD-HMMs are used to model continuous valued observations
- State is modeled as a continuous random variable
- Eg. Emission probability of CD-HMMs can be modeled using Gaussian

$$b_j(\mathbf{v}) \sim \mathcal{N}(\mathbf{v}/\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- In speech processing, the state is modeled as a M-mixture GMM

$$b_j(\mathbf{v}) = \sum_{k=1}^{M} w_{jk} \mathcal{N}(\mathbf{v}/\boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk})$$

Continuous Density HMMs

- CD-HMMs are used to model continuous valued observations
- State is modeled as a continuous random variable
- Eg. Emission probability of CD-HMMs can be modeled using Gaussian

$$b_j(\mathbf{v}) \sim \mathcal{N}(\mathbf{v}/\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

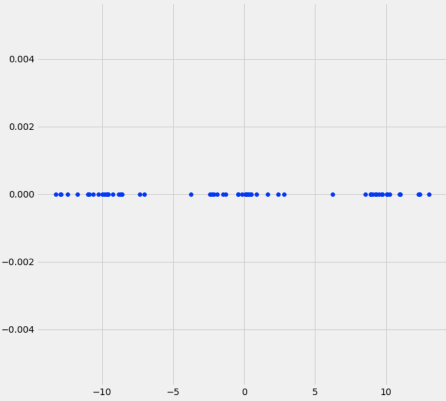
- In speech processing, the state is modeled as a M-mixture GMM

$$b_j(\mathbf{v}) = \sum_{k=1}^{M} w_{jk} \mathcal{N}(\mathbf{v}/\boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk})$$

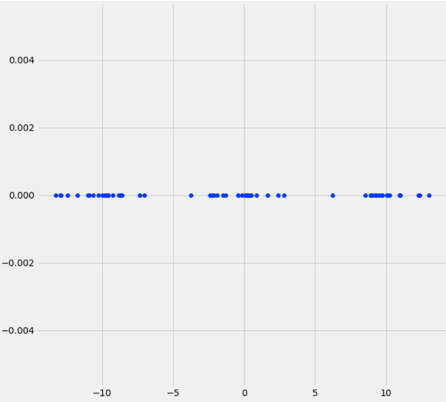
- In order to ensure that the pdf of the state integrates to 1

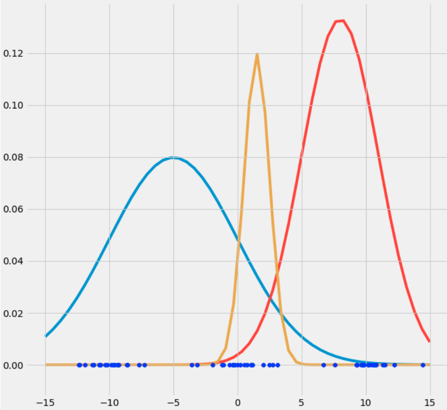
$$\sum_{k=1}^{M} w_{jk} = 1$$

When one Gaussian is not enough!



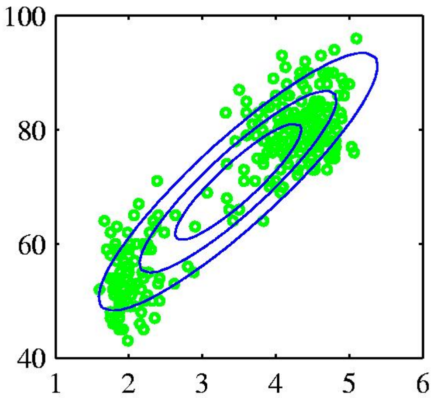
When one Gaussian is not enough!



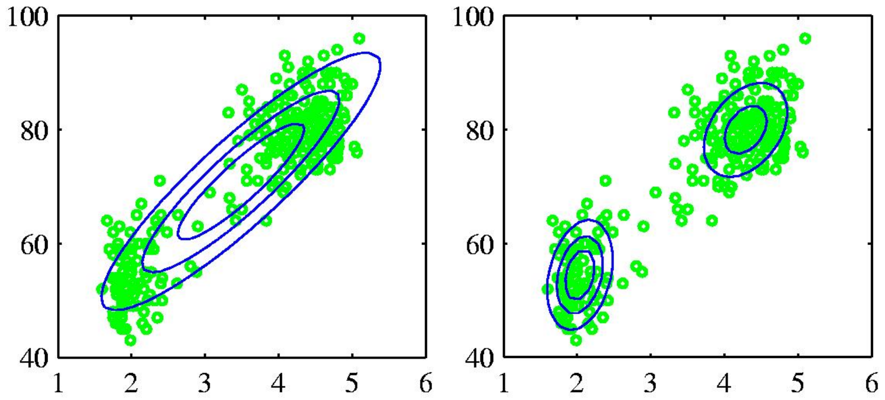


- Real-world datasets can have multimodal distribution
- Single Gaussian may not be a good approximation

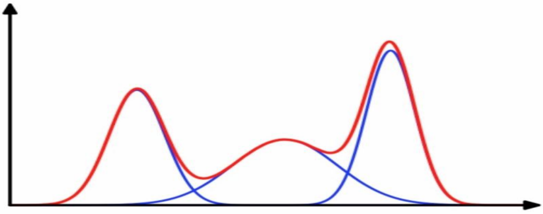
2-D Illustration



2-D Illustration

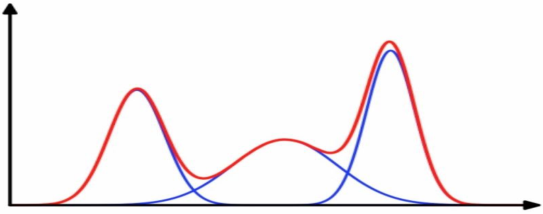


Gaussian Mixture Model (GMM)



- PDF can be written as a linear weighted sum of Gaussian distributions

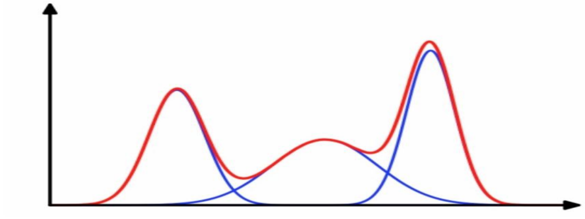
Gaussian Mixture Model (GMM)



- PDF can be written as a linear weighted sum of Gaussian distributions
- Linear superposition of Gaussians

$$p(x) = \sum_{m=1}^{M} \pi_m \mathcal{N}(x/\mu_m, \sigma_m^2)$$

Gaussian Mixture Model (GMM)



- PDF can be written as a linear weighted sum of Gaussian distributions
- Linear superposition of Gaussians

$$p(x) = \sum_{m=1}^{M} \pi_m \mathcal{N}(x/\mu_m, \sigma_m^2)$$

- PDF should be positive and integrate to one

$$\pi_m \geq 0 \qquad \& \qquad \sum^{M} \pi_m = 1$$

Sampling from GMM

- Parameters of the GMM $(\pi_m, \mu_m, \sigma_m^2), \quad m = 1, 2, \cdots, M$

Sampling from GMM

- Parameters of the GMM $(\pi_m, \mu_m, \sigma_m^2), \quad m = 1, 2, \cdots, M$
- Data generation process
  - Roll an M-sided dice and observe the outcome (side k)
  - Generate a datapoint from the k^{th} Gaussian component (μ_k, σ_k^2)
  - Repeat the above two steps for generate multiple datapoints

Sampling from GMM

- Parameters of the GMM $(\pi_m, \mu_m, \sigma_m^2), \quad m = 1, 2, \cdots, M$
- Data generation process
  - Roll an M-sided dice and observe the outcome (side k)
  - Generate a datapoint from the k^{th} Gaussian component (μ_k, σ_k^2)
  - Repeat the above two steps for generate multiple datapoints
- Given the observed data, we need to estimate the underlying pdf
  - Number of Gaussian components M is not known
  - M is typically determined empirically
  - Parameters of the GMM are estimated by maximizing the ML criterion

ML Formulation for GMM

ML Formulation for GMM

- Consider data $X = \{x_1, x_2, \cdots x_N\}$ drawn from unknown distribution

ML Formulation for GMM

- Consider data $X = \{x_1, x_2, \cdots x_N\}$ drawn from unknown distribution
- Let the unknown distribution $p(x_n)$ be approximated by a GMM

$$p(x_n) = \sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)$$

ML Formulation for GMM

- Consider data $X = \{x_1, x_2, \cdots x_N\}$ drawn from unknown distribution
- Let the unknown distribution $p(x_n)$ be approximated by a GMM

$$p(x_n) = \sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)$$

- Assuming i.i.d sampling, the likelihood function can expressed as

$$\mathcal{L}(\theta/X) = P(X/\theta) = \prod_{n=1}^{N} \sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)$$

ML Formulation for GMM

- Consider data $X = \{x_1, x_2, \cdots x_N\}$ drawn from unknown distribution
- Let the unknown distribution $p(x_n)$ be approximated by a GMM

$$p(x_n) = \sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)$$

- Assuming i.i.d sampling, the likelihood function can expressed as

$$\mathcal{L}(\theta/X) = P(X/\theta) = \prod_{n=1}^{N} \sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)$$

- GMM parameters can be estimated by maximizing the log-likelihood

$$J(\theta) = \sum_{n=1}^{N} \log \left(\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2) \right) + \lambda \sum_{m=1}^{M} \pi_m = 1$$

Estimating μ_k

$$J(\theta) = \sum_{n=1}^{N} \log \left(\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2) \right) + \lambda \sum_{m=1}^{M} \pi_m = 1$$

- Taking partial derivative w.r.t μ_k

$$\frac{\partial J(\theta)}{\partial \mu_k} \quad =$$

Estimating μ_k

$$J(\theta) = \sum_{n=1}^{N} \log \left(\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2) \right) + \lambda \sum_{m=1}^{M} \pi_m = 1$$

- Taking partial derivative w.r.t μ_k

$$\frac{\partial J(\theta)}{\partial \mu_k} = \sum_{n=1}^{N}$$

Estimating μ_k

$$J(\theta) = \sum_{n=1}^{N} \log\left(\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)\right) + \lambda \sum_{m=1}^{M} \pi_m = 1$$

- Taking partial derivative w.r.t μ_k

$$\frac{\partial J(\theta)}{\partial \mu_k} = \sum_{n=1}^{N} \frac{1}{\sum\limits_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)}$$

Estimating μ_k

$$J(\theta) = \sum_{n=1}^{N} \log \left(\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2) \right) + \lambda \sum_{m=1}^{M} \pi_m = 1$$

- Taking partial derivative w.r.t μ_k

$$\frac{\partial J(\theta)}{\partial \mu_k} = \sum_{n=1}^{N} \frac{1}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)} \pi_k \frac{\partial}{\partial \mu_k} \mathcal{N}(x_n/\mu_k, \sigma_k^2)$$

Estimating μ_k

$$J(\theta) = \sum_{n=1}^{N} \log\left(\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)\right) + \lambda \sum_{m=1}^{M} \pi_m = 1$$

- Taking partial derivative w.r.t μ_k

$$\begin{aligned}
\frac{\partial J(\theta)}{\partial \mu_k} &= \sum_{n=1}^{N} \frac{1}{\sum\limits_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)} \pi_k \frac{\partial}{\partial \mu_k} \mathcal{N}(x_n/\mu_k, \sigma_k^2) \\
&= \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum\limits_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)} \left(\frac{x_n - \mu_k}{\sigma_k^2}\right)
\end{aligned}$$

Estimating μ_k

$$J(\theta) = \sum_{n=1}^{N} \log \left(\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2) \right) + \lambda \sum_{m=1}^{M} \pi_m = 1$$

- Taking partial derivative w.r.t μ_k

$$\begin{aligned}
\frac{\partial J(\theta)}{\partial \mu_k} &= \sum_{n=1}^{N} \frac{1}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)} \pi_k \frac{\partial}{\partial \mu_k} \mathcal{N}(x_n/\mu_k, \sigma_k^2) \\
&= \sum_{n=1}^{N} \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)} \left(\frac{x_n - \mu_k}{\sigma_k^2} \right) \\
&= \sum_{n=1}^{N} \gamma_{nk} \left(\frac{x_n - \mu_k}{\sigma_k^2} \right)
\end{aligned}$$

Interpretation of γ_{nk}

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)}$$

Interpretation of γ_{nk}

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum\limits_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)}$$

Interpretation of γ_{nk}

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)}$$

- γ_{nk}: Responsibility of k^{th} Gaussain in generating n^{th} datapoint

Interpretation of γ_{nk}

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)}$$

- γ_{nk}: Responsibility of k^{th} Gaussain in generating n^{th} datapoint
- All components should collectively share the responsibility

$$\sum_{k=1}^{M} \gamma_{nk} = 1$$

Interpretation of γ_{nk}

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)}$$

- γ_{nk}: Responsibility of k^{th} Gaussain in generating n^{th} datapoint
- All components should collectively share the responsibility

$$\sum_{k=1}^{M} \gamma_{nk} = 1$$

- Effective number of points generated by k^{th} component

$$\sum_{n=1}^{N} \gamma_{nk} = N_k$$

Interpretation of γ_{nk}

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)}$$

- γ_{nk}: Responsibility of k^{th} Gaussain in generating n^{th} datapoint
- All components should collectively share the responsibility

$$\sum_{k=1}^{M} \gamma_{nk} = 1$$

- Effective number of points generated by k^{th} component

$$\sum_{n=1}^{N} \gamma_{nk} = N_k \qquad \sum_{k=1}^{M} N_k = N$$

Interpretation of γ_{nk}

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)}$$

- γ_{nk}: Responsibility of k^{th} Gaussain in generating n^{th} datapoint
- All components should collectively share the responsibility

$$\sum_{k=1}^{M} \gamma_{nk} = 1$$

- Effective number of points generated by k^{th} component

$$\sum_{n=1}^{N} \gamma_{nk} = N_k \qquad \sum_{k=1}^{M} N_k = N$$

- After estimation, γ_{nk} can be interpreted as posterior probability

Estimates of GMM Parameters

Estimates of GMM Parameters

- Estimated mean of the k^{th} component

$$\hat{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} x_n \qquad k = 1, 2, \cdots M$$

Estimates of GMM Parameters

- Estimated mean of the k^{th} component

$$\hat{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma_{nk}x_n \qquad k = 1, 2, \cdots M$$

- Estimated variance of the k^{th} component

$$\hat{\sigma}_k^2 = \frac{1}{N_k}\sum_{n=1}^{N}\gamma_{nk}(x_n - \hat{\mu}_k)^2 \qquad k = 1, 2, \cdots M$$

Estimates of GMM Parameters

- Estimated mean of the k^{th} component

$$\hat{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N} \gamma_{nk} x_n \qquad k = 1, 2, \cdots M$$

- Estimated variance of the k^{th} component

$$\hat{\sigma}_k^2 = \frac{1}{N_k}\sum_{n=1}^{N} \gamma_{nk} (x_n - \hat{\mu}_k)^2 \qquad k = 1, 2, \cdots M$$

- Estimated weight of the k^{th} component (prior)

$$\hat{\pi}_k = \frac{N_k}{N} \qquad k = 1, 2, \cdots M$$

Estimates of GMM Parameters

- Estimated mean of the k^{th} component

$$\hat{\mu}_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma_{nk}x_n \qquad k = 1, 2, \cdots M$$

- Estimated variance of the k^{th} component

$$\hat{\sigma}_k^2 = \frac{1}{N_k}\sum_{n=1}^{N}\gamma_{nk}(x_n - \hat{\mu}_k)^2 \qquad k = 1, 2, \cdots M$$

- Estimated weight of the k^{th} component (prior)

$$\hat{\pi}_k = \frac{N_k}{N} \qquad k = 1, 2, \cdots M$$

- Are they in closed form? Can we directly compute them from data?

EM Algorithm for GMM

EM Algorithm for GMM

- Given data $X = \{x_1, x_2, \cdots x_N\}$

EM Algorithm for GMM

- Given data $X = \{x_1, x_2, \cdots x_N\}$
- **Initialization**: Choose the number of mixtures M

EM Algorithm for GMM

- Given data $X = \{x_1, x_2, \cdots x_N\}$
- **Initialization**: Choose the number of mixtures M
  - Randomly choose means (μ_k), variances (σ_k^2) and mixture weights (π_k)

EM Algorithm for GMM

- Given data $X = \{x_1, x_2, \cdots x_N\}$
- **Initialization**: Choose the number of mixtures M
  - Randomly choose means (μ_k), variances (σ_k^2) and mixture weights (π_k)
  - Evaluate log-likelihood with initial conditions $p(X/\theta)$

EM Algorithm for GMM

- Given data $X = \{x_1, x_2, \cdots x_N\}$
- **Initialization**: Choose the number of mixtures M
  - Randomly choose means (μ_k), variances (σ_k^2) and mixture weights (π_k)
  - Evaluate log-likelihood with initial conditions $p(X/\theta)$
- **Expectation step**: Evaluate the responsibilities using the current θ

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)}$$

EM Algorithm for GMM

- Given data $X = \{x_1, x_2, \cdots x_N\}$
- **Initialization**: Choose the number of mixtures M
  - Randomly choose means (μ_k), variances (σ_k^2) and mixture weights (π_k)
  - Evaluate log-likelihood with initial conditions $p(X/\theta)$
- **Expectation step**: Evaluate the responsibilities using the current θ

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)} \qquad N_k = \sum_{n=1}^{N} \gamma_{nk}$$

EM Algorithm for GMM

- Given data $X = \{x_1, x_2, \cdots x_N\}$
- **Initialization**: Choose the number of mixtures M
  - Randomly choose means (μ_k), variances (σ_k^2) and mixture weights (π_k)
  - Evaluate log-likelihood with initial conditions $p(X/\theta)$
- **Expectation step**: Evaluate the responsibilities using the current θ

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)} \qquad N_k = \sum_{n=1}^{N} \gamma_{nk}$$

- **Maximization step**: Update parameters with current γ_{nk}

$$\hat{\mu}_k^{new} = \frac{1}{N_k} \qquad \hat{\sigma}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk} (x_n - \hat{\mu}_k)^2 \qquad \hat{\pi}_k^{new} = \frac{N_k}{N}$$

EM Algorithm for GMM

- Given data $X = \{x_1, x_2, \cdots x_N\}$
- **Initialization**: Choose the number of mixtures M
  - Randomly choose means (μ_k), variances (σ_k^2) and mixture weights (π_k)
  - Evaluate log-likelihood with initial conditions $p(X/\theta)$
- **Expectation step**: Evaluate the responsibilities using the current θ

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)} \qquad N_k = \sum_{n=1}^{N} \gamma_{nk}$$

- **Maximization step**: Update parameters with current γ_{nk}

$$\hat{\mu}_k^{new} = \frac{1}{N_k} \qquad \hat{\sigma}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk}(x_n - \hat{\mu}_k)^2 \qquad \hat{\pi}_k^{new} = \frac{N_k}{N}$$

- Compute log-likelihood with updated parameters

EM Algorithm for GMM

- Given data $X = \{x_1, x_2, \cdots x_N\}$
- **Initialization**: Choose the number of mixtures M
  - Randomly choose means (μ_k), variances (σ_k^2) and mixture weights (π_k)
  - Evaluate log-likelihood with initial conditions $p(X/\theta)$
- **Expectation step**: Evaluate the responsibilities using the current θ

$$\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n/\mu_k, \sigma_k^2)}{\sum_{m=1}^{M} \pi_m \mathcal{N}(x_n/\mu_m, \sigma_m^2)} \qquad N_k = \sum_{n=1}^{N} \gamma_{nk}$$

- **Maximization step**: Update parameters with current γ_{nk}

$$\hat{\mu}_k^{new} = \frac{1}{N_k} \qquad \hat{\sigma}_k^{new} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{nk}(x_n - \hat{\mu}_k)^2 \qquad \hat{\pi}_k^{new} = \frac{N_k}{N}$$

- Compute log-likelihood with updated parameters
- Repeat E-step and M-step until convergence

Illustration of EM Iterations

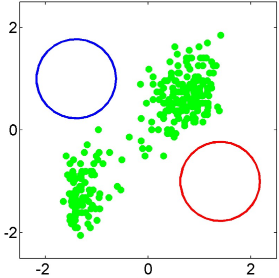


Illustration of EM Iterations

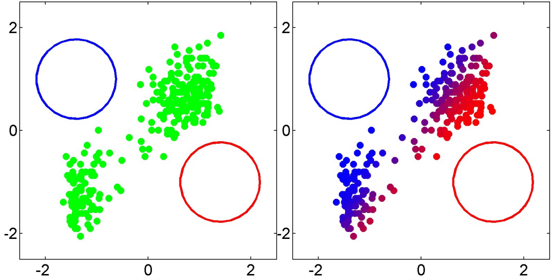


Illustration of EM Iterations

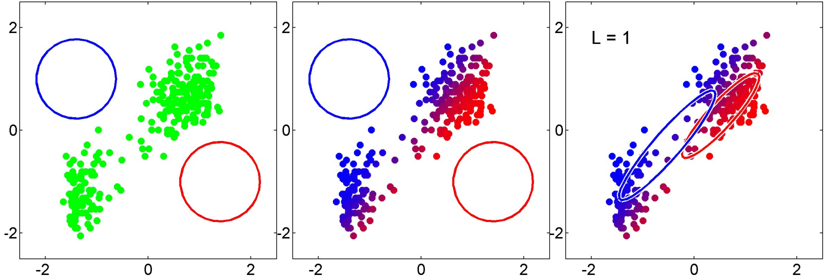


Illustration of EM Iterations

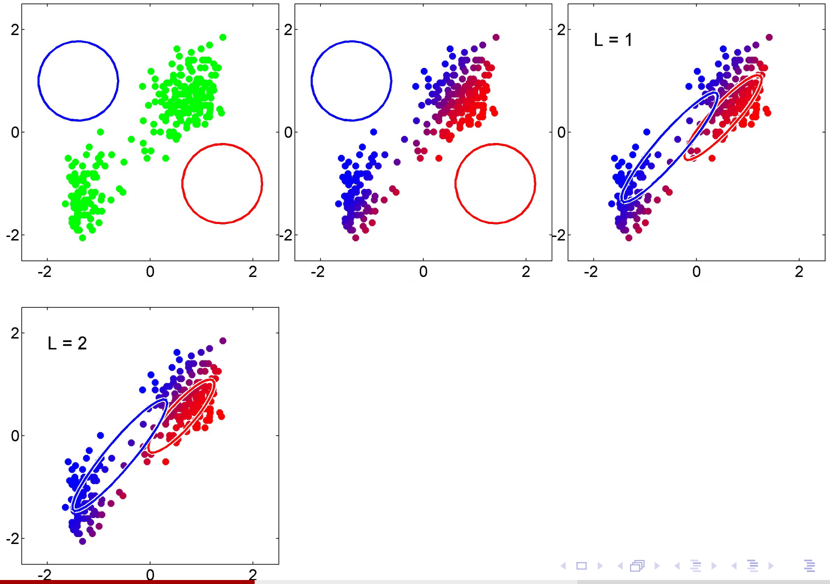


Illustration of EM Iterations

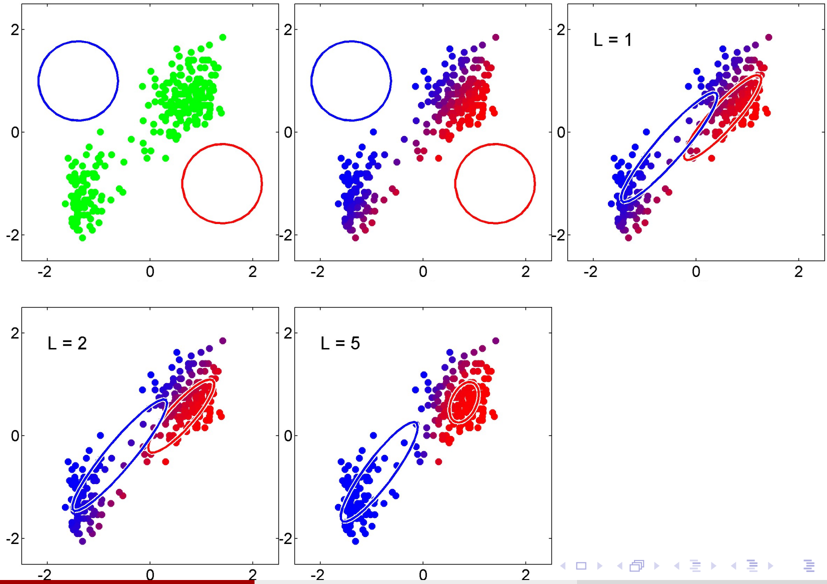


Illustration of EM Iterations

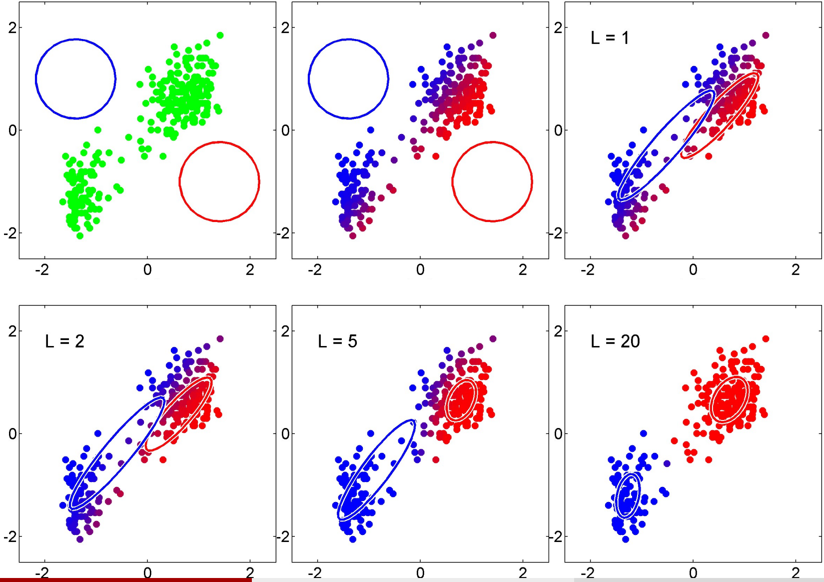
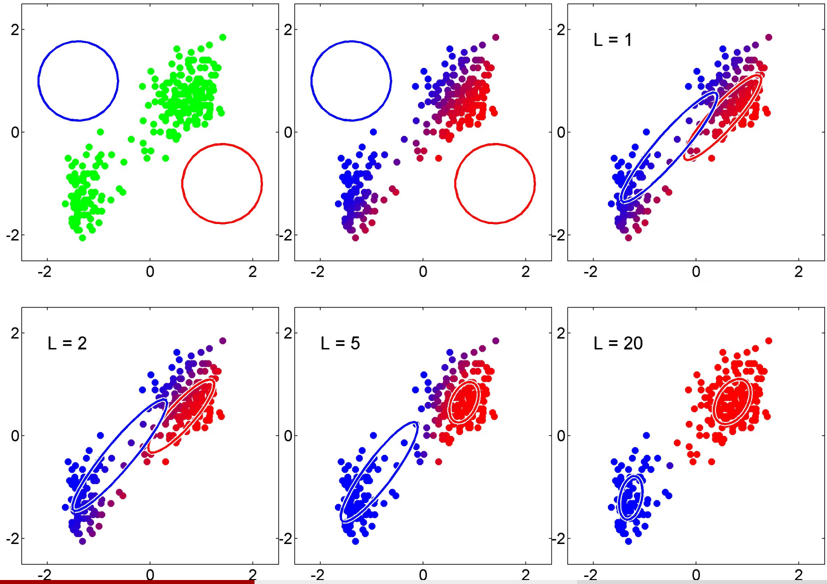
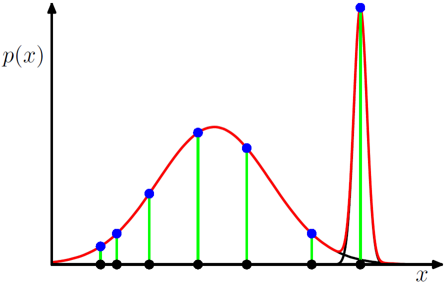


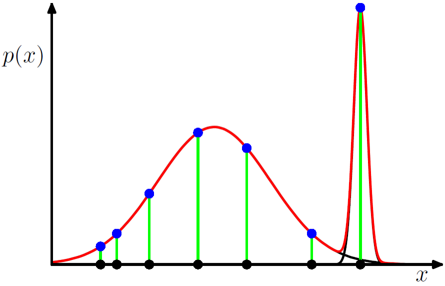
Illustration of EM Iterations



Singularities in Likelihood Function

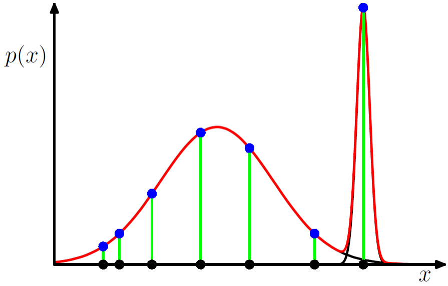


Singularities in Likelihood Function



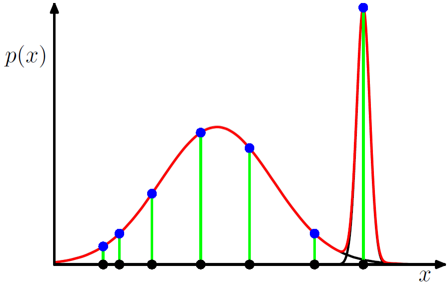
- One component has finite variance

Singularities in Likelihood Function



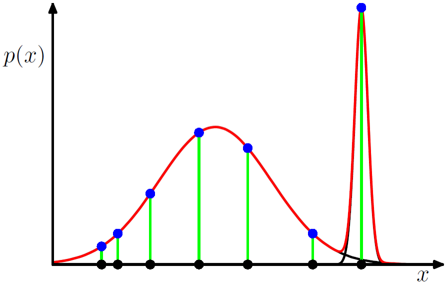
- One component has finite variance
- Some Gaussian components can collapse to specific data points

Singularities in Likelihood Function



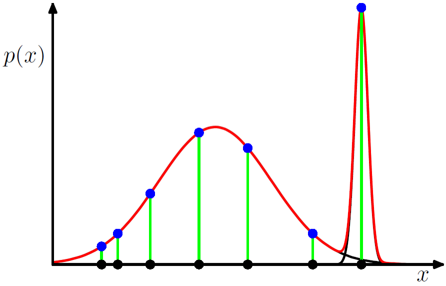
- One component has finite variance
- Some Gaussian components can collapse to specific data points
- Shrinking variance can lead to ever increasing log-likelihood!

Singularities in Likelihood Function



- One component has finite variance
- Some Gaussian components can collapse to specific data points
- Shrinking variance can lead to ever increasing log-likelihood!
- ML approach can result in severe over-fitting (local maximum)

Singularities in Likelihood Function



- One component has finite variance
- Some Gaussian components can collapse to specific data points
- Shrinking variance can lead to ever increasing log-likelihood!
- ML approach can result in severe over-fitting (local maximum)
- Poor initialization, outliers, smaller dataset, more components