Hidden Markov Models. Applications in Bioinformatics

Javier Campos

Universidad de Zaragoza

http://webdiis.unizar.es/~jcampos/

Adapted and expanded from slides by Andrew W. Moore (http://www.cs.cmu.edu/~awm)

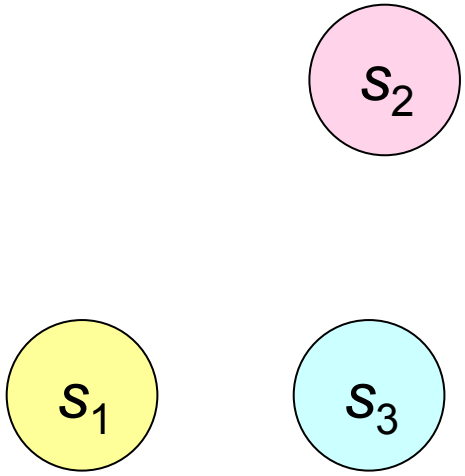
Outline

- Brief introduction to Markov models
- Hidden Markov Models
- Three typical problems on HMMs:
  - Evaluation → forward-backward algorithms
  - Inference → Viterbi decoding algorithm
  - Learning → Baum–Welch (Expectation Maximization) algorithm
- Applications in Bioinformatics
  - Segmentation of biological sequences
  - Multiple alignment of biological sequences
  - Case study (reading matter): odorant receptors

A Markov System

Has N states, called s_1, s_2 …s_N

There are discrete timesteps, t=0, t=1…



$N = 3$

$t = 0$

A Markov System

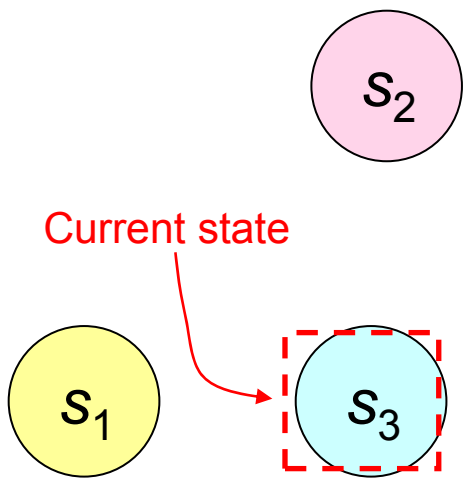
Has N states, called s_1, s_2 …s_N

There are discrete timesteps, t=0, t=1…

On the t'th timestep the system is in exactly one of the available states.

Call it q_t

Note: $q_t \in \{s_1, s_2 \ldots s_N\}$

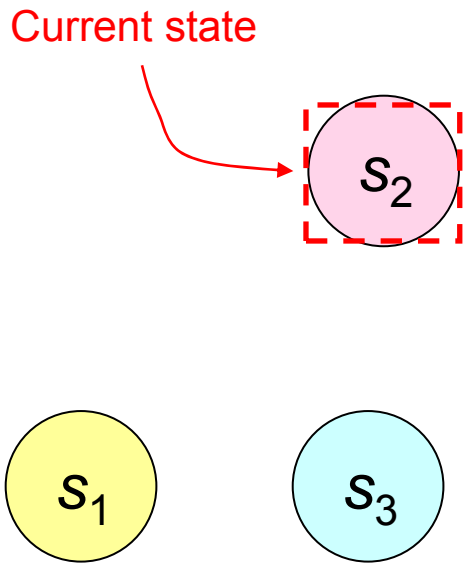


$N = 3$

$t = 0$

$q_t = q_0 = s_3$

A Markov System



$N = 3$

$t = 0$

$q_t = q_1 = s_2$

Has N states, called s_1, $s_2 \ldots s_N$

There are discrete timesteps, t=0, t=1…

On the t'th timestep the system is in exactly one of the available states.

Call it q_t

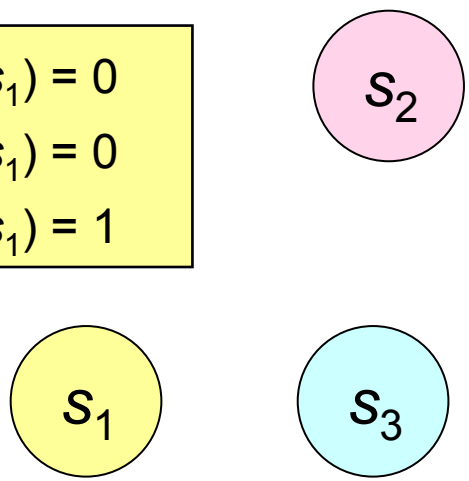
Note: $q_t \in \{s_1, s_2 \ldots s_N\}$

Between each time step, the next state is chosen randomly.

A Markov System

$P(q_{t+1}=s_1|q_t=s_2) = 1/2$
$P(q_{t+1}=s_2|q_t=s_2) = 1/2$
$P(q_{t+1}=s_3|q_t=s_2) = 0$

$P(q_{t+1}=s_1|q_t=s_1) = 0$
$P(q_{t+1}=s_2|q_t=s_1) = 0$
$P(q_{t+1}=s_3|q_t=s_1) = 1$



$N = 3$

$t = 0$

$q_t=q_1=s_2$

$P(q_{t+1}=s_1|q_t=s_3) = 1/3$
$P(q_{t+1}=s_2|q_t=s_3) = 2/3$
$P(q_{t+1}=s_3|q_t=s_3) = 0$

Has N states, called s_1, $s_2 \ldots s_N$

There are discrete timesteps, t=0, t=1…

On the t'th timestep the system is in exactly one of the available states.

Call it q_t

Note: $q_t \in \{s_1, s_2 \ldots s_N\}$

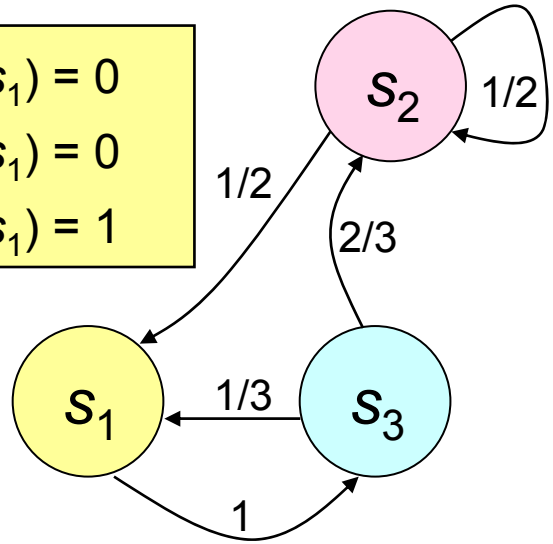
Between each time step, the next state is chosen randomly.

The current state determines the probability distribution for the next state.

A Markov System

$P(q_{t+1}=s_1|q_t=s_2) = 1/2$
$P(q_{t+1}=s_2|q_t=s_2) = 1/2$
$P(q_{t+1}=s_3|q_t=s_2) = 0$

$P(q_{t+1}=s_1|q_t=s_1) = 0$
$P(q_{t+1}=s_2|q_t=s_1) = 0$
$P(q_{t+1}=s_3|q_t=s_1) = 1$



$N = 3$

$t = 0$

$q_t=q_1=s_2$

$P(q_{t+1}=s_1|q_t=s_3) = 1/3$
$P(q_{t+1}=s_2|q_t=s_3) = 2/3$
$P(q_{t+1}=s_3|q_t=s_3) = 0$

Often notated with arcs between states

Has N states, called s_1, s_2 …s_N

There are discrete timesteps, t=0, t=1…

On the t'th timestep the system is in exactly one of the available states.

Call it q_t

Note: $q_t \in \{s_1, s_2 \ldots s_N\}$

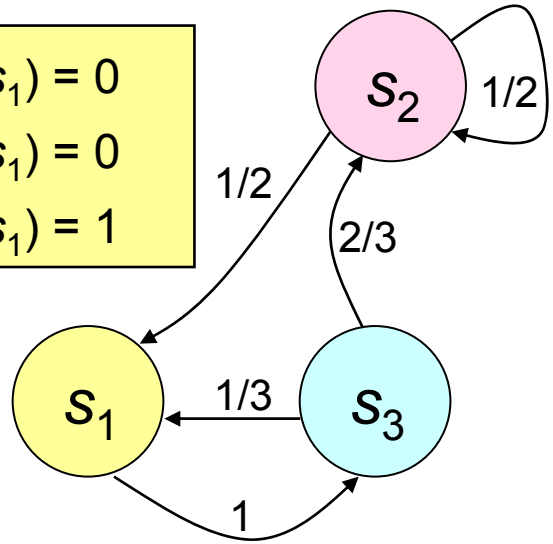
Between each time step, the next state is chosen randomly.

The current state determines the probability distribution for the next state.

A Markov System

$P(q_{t+1}=s_1|q_t=s_2) = 1/2$
$P(q_{t+1}=s_2|q_t=s_2) = 1/2$
$P(q_{t+1}=s_3|q_t=s_2) = 0$

$P(q_{t+1}=s_1|q_t=s_1) = 0$
$P(q_{t+1}=s_2|q_t=s_1) = 0$
$P(q_{t+1}=s_3|q_t=s_1) = 1$



$N = 3$

$t = 0$

$q_t=q_1=s_2$

$P(q_{t+1}=s_1|q_t=s_3) = 1/3$
$P(q_{t+1}=s_2|q_t=s_3) = 2/3$
$P(q_{t+1}=s_3|q_t=s_3) = 0$

q_{t+1} is conditionally independent of $\{q_{t-1}, q_{t-2} \ldots q_1, q_0\}$ given q_t.

In other words:

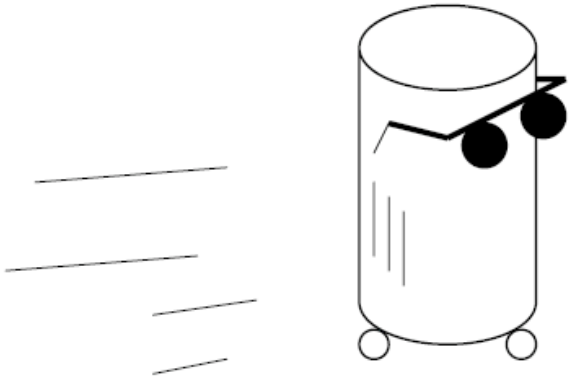
$P(q_{t+1}=s_j \mid q_t=s_i) =$

$P(q_{t+1}=s_j \mid q_t=s_i$, any earlier history$)$

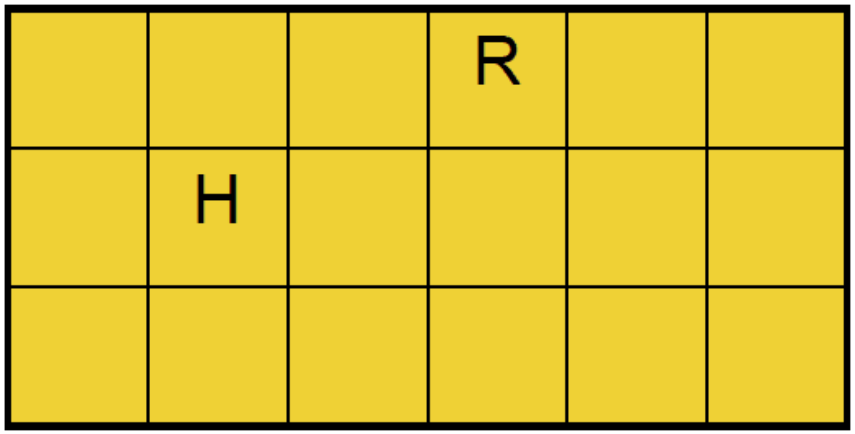
Notation:

$a_{ij} = P(q_{t+1}=s_j \mid q_t=s_i)$

A Blind Robot



A human and a robot wander around randomly on a grid…

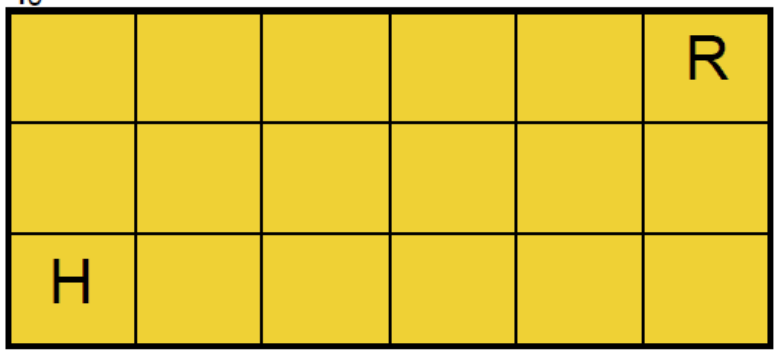


STATE q = ⟨ Location of Robot, Location of Human ⟩

Note: N (num. states) = 18 * 18 = 324

Dynamics of System

q_0 =



Each timestep the human moves randomly to an adjacent cell. And Robot also moves randomly to an adjacent cell.

Typical Questions:

- "What's the expected time until the human is crushed like a bug?"
- "What's the probability that the robot will hit the left wall before it hits the human?"
- "What's the probability Robot crushes human on next time step?"

Example Question

"It's currently time t, and human remains uncrushed. What's the probability of crushing occurring at time t + 1 ?"

If robot is blind:

We can compute this in advance.

We'll do this first

If robot is omnipotent:

(I.E. If robot knows state at time t), can compute directly.

Too Easy. We won't do this

If robot has some sensors, but incomplete state information …

Hidden Markov Models are applicable!

Main Body of Lecture

What is $P(q_t = s)$? slow, stupid answer

Step 1: Work out how to compute P(Q) for any path Q $= q_1\ q_2\ q_3 \ldots q_t$

Given we know the start state q_1 (i.e. $P(q_1)=1$)

$$P(q_1\ q_2 \ldots q_t) = P(q_1\ q_2 \ldots q_{t-1})\ P(q_t | q_1\ q_2 \ldots q_{t-1})$$

$$= P(q_1\ q_2 \ldots q_{t-1})\ P(q_t | q_{t-1})$$ *WHY?*

$$= P(q_2|q_1)P(q_3|q_2)\ldots P(q_t|q_{t-1})$$

Step 2: Use this knowledge to get $P(q_t = s)$

$$P(q_t = s) = \sum_{Q \in \text{Paths of length } t \text{ that end in } s} P(Q)$$

Computation is exponential in t

What is $P(q_t = s)$? Clever answer

- For each state s_i, define

  $p_t(i)$ = Prob. state is s_i at time t

  $= P(q_t = s_i)$

- Easy to do inductive definition

$$\forall i \quad p_0(i) =$$

$$\forall j \quad p_{t+1}(j) = P(q_{t+1} = s_j) =$$

What is $P(q_t = s)$? Clever answer

- For each state s_i, define

  $p_t(i)$ = Prob. state is s_i at time t

  $= P(q_t = s_i)$

- Easy to do inductive definition

$$\forall i \quad p_0(i) = \begin{cases} 1 & \text{if } s_i \text{ is the start state} \\ 0 & \text{otherwise} \end{cases}$$

$$\forall j \quad p_{t+1}(j) = P(q_{t+1} = s_j) =$$

What is $P(q_t = s)$? Clever answer

- For each state s_i, define

  $p_t(i)$ = Prob. state is s_i at time t

  $= P(q_t = s_i)$

- Easy to do inductive definition

$$\forall i \quad p_0(i) = \begin{cases} 1 & \text{if } s_i \text{ is the start state} \\ 0 & \text{otherwise} \end{cases}$$

$$\forall j \quad p_{t+1}(j) = P(q_{t+1} = s_j) =$$

$$\sum_{i=1}^{N} P(q_{t+1} = s_j \wedge q_t = s_i) =$$

What is $P(q_t = s)$? Clever answer

- For each state s_i, define

  $p_t(i)$ = Prob. state is s_i at time t

  $= P(q_t = s_i)$

- Easy to do inductive definition

$$\forall i \quad p_0(i) = \begin{cases} 1 & \text{if } s_i \text{ is the start state} \\ 0 & \text{otherwise} \end{cases}$$

$$\forall j \quad p_{t+1}(j) = P(q_{t+1} = s_j) =$$

$$\sum_{i=1}^{N} P(q_{t+1} = s_j \wedge q_t = s_i) =$$

$$\sum_{i=1}^{N} P(q_{t+1} = s_j \mid q_t = s_i) P(q_t = s_i) = \sum_{i=1}^{N} a_{ij} p_t(i)$$

Remember,

$$a_{ij} = P(q_{t+1} = s_j \mid q_t = s_i)$$

What is $P(q_t = s)$? Clever answer

- For each state s_i, define

  $p_t(i)$ = Prob. state is s_i at time t

  $= P(q_t = s_i)$

- Easy to do inductive definition

$$\forall i \quad p_0(i) = \begin{cases} 1 & \text{if } s_i \text{ is the start state} \\ 0 & \text{otherwise} \end{cases}$$

$$\forall j \quad p_{t+1}(j) = P(q_{t+1} = s_j) =$$

$$\sum_{i=1}^{N} P(q_{t+1} = s_j \wedge q_t = s_i) =$$

$$\sum_{i=1}^{N} P(q_{t+1} = s_j \mid q_t = s_i) P(q_t = s_i) = \sum_{i=1}^{N} a_{ij} p_t(i)$$

- Computation is simple.
- Just fill in this table in this order:

t	$p_t(1)$	$p_t(2)$	…	$p_t(N)$
0	0	1		0
1				
:				
t_{final}				

What is $P(q_t = s)$? Clever answer

- For each state s_i, define

  $p_t(i)$ = Prob. state is s_i at time t

  $= P(q_t = s_i)$

- Easy to do inductive definition

$$\forall i \quad p_0(i) = \begin{cases} 1 & \text{if } s_i \text{ is the start state} \\ 0 & \text{otherwise} \end{cases}$$

$$\forall j \quad p_{t+1}(j) = P(q_{t+1} = s_j) =$$

$$\sum_{i=1}^{N} P(q_{t+1} = s_j \wedge q_t = s_i) =$$

$$\sum_{i=1}^{N} P(q_{t+1} = s_j \mid q_t = s_i) P(q_t = s_i) = \sum_{i=1}^{N} a_{ij} p_t(i)$$

- Cost of computing $P_t(i)$ for all states S_i is now $O(t\,N^2)$
- The stupid way was $O(N^t)$
- This was a simple example
- It was meant to warm you up to this trick, called *Dynamic Programming*, because HMMs do many tricks like this.(*)

(*) Read the basics on *Dynamic Programming* (D.P.) here (in Spanish):
http://webdiis.unizar.es/asignaturas/EDA/ea/slides/4-Programacion%20dinamica.pdf

Outline

- Brief introduction to Markov models
- Hidden Markov Models
- Three typical problems on HMMs:
  - Evaluation → forward-backward algorithms
  - Inference → Viterbi decoding algorithm
  - Learning → Baum–Welch (Expectation Maximization) algorithm
- Applications in Bioinformatics
  - Segmentation of biological sequences
  - Multiple alignment of biological sequences
  - Case study (reading matter): odorant receptors

Hidden State

"It's currently time t, and human remains uncrushed. What's the probability of crushing occurring at time t + 1 ?"

If robot is blind:

We can compute this in advance.

We'll do this first

If robot is omnipotent:

(I.E. If robot knows state at time t), can compute directly.

Too Easy. We won't do this

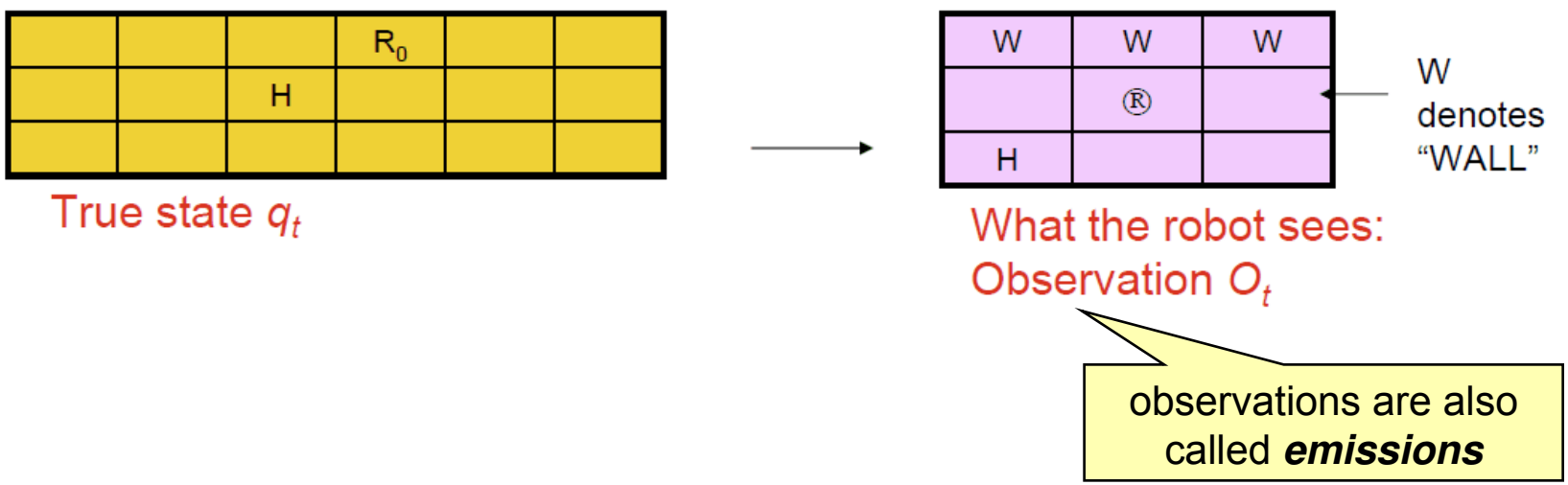
If robot has some sensors, but incomplete state information …

Hidden Markov Models are applicable!

Main Body of Lecture

Hidden State

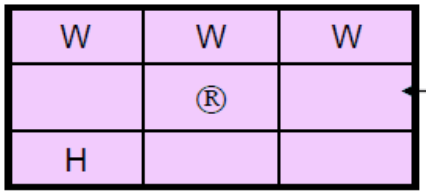
- The previous example tried to estimate $P(q_t = s_i)$ unconditionally (using no observed evidence).
- Suppose we can observe something that's affected by the true state.
- Example: Proximity sensors. (tell us the contents of the 8 adjacent squares)



Noisy Hidden State

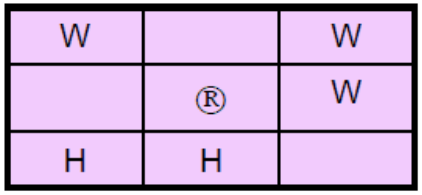
- Example: Noisy Proximity sensors. (unreliably tell us the contents of the 8 adjacent squares)

True state q_t



W denotes "WALL"

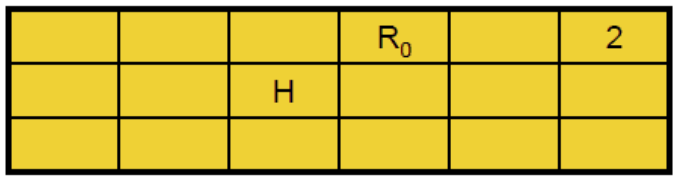
Uncorrupted Observation



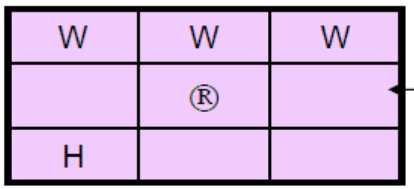
What the robot sees: Observation O_t

Noisy Hidden State

- Example: Noisy Proximity sensors. (unreliably tell us the contents of the 8 adjacent squares)

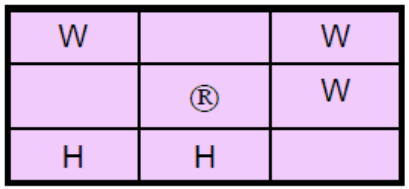


True state q_t



W denotes "WALL"

Uncorrupted Observation



What the robot sees: Observation O_t

O_t is noisily determined depending on the current state.

Assume that O_t is conditionally independent of $\{q_{t-1}, q_{t-2}, \ldots q_1, q_0, O_{t-1}, O_{t-2}, \ldots O_1, O_0\}$ given q_t.

In other words:

$P(O_t = X \mid q_t = s_i) =$

$P(O_t = X \mid q_t = s_i, \text{any earlier history})$

Notation:

$b_i(k) = P(O_t = k \mid q_t = s_i)$

Hidden Markov Models

Our robot with noisy sensors is a good example of an HMM

- Question 1: State Estimation

  What is $P(q_T=S_i \mid O_1O_2\ldots O_T)$

  It will turn out that a new cute D.P. trick will get this for us.

- Question 2: Most Probable Path

  Given $O_1O_2\ldots O_T$, what is the most probable path that I took?

  And what is that probability?

  Yet another famous D.P. trick, the VITERBI algorithm, gets this.

- Question 3: Learning HMMs:

  Given $O_1O_2\ldots O_T$, what is the maximum likelihood HMM that could have produced this string of observations?

  Very very useful. Uses the E.M. Algorithm

Expectation Maximization

Are H.M.M.s Useful?

- Robot planning + sensing when there's uncertainty
- Speech Recognition / Understanding
- Consumer decision modeling
- Economics & Finance

… i.e. complicated stuff your lecturer knows nothing about.

- Bioinformatics
  - Segmentation (define regions' boundaries in gene & protein sequences)
  - Alignment of biological sequences
  - Gene finding

- Plus at least 5 other things I haven't thought of.

Outline

- Brief introduction to Markov models
- Hidden Markov Models
- Three typical problems on HMMs:
  - Evaluation → forward-backward algorithms
  - Inference → Viterbi decoding algorithm
  - Learning → Baum–Welch (Expectation Maximization) algorithm
- Applications in Bioinformatics
  - Segmentation of biological sequences
  - Multiple alignment of biological sequences
  - Case study (reading matter): odorant receptors

Some Famous HMM Tasks

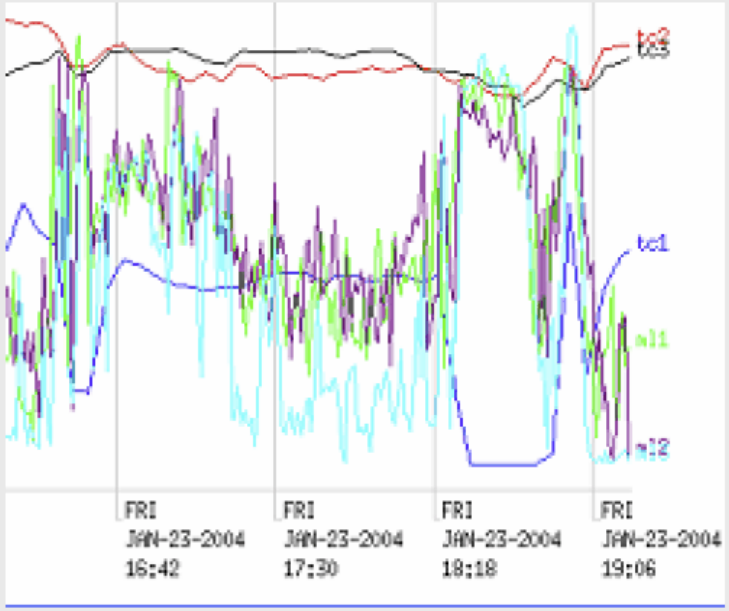
Question 1: State Estimation

What is $P(q_T=S_i \mid O_1O_2\ldots O_t)$

Some Famous HMM Tasks

Question 1: State Estimation

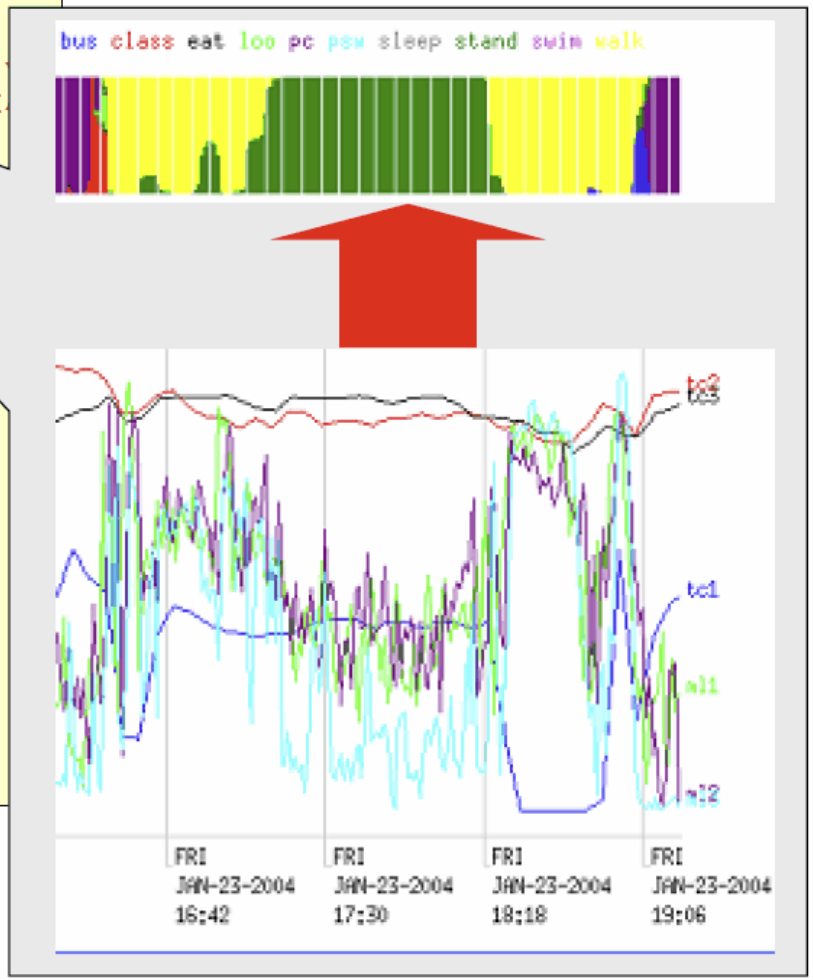
What is $P(q_T = S_i | O_1 O_2 \ldots O_t)$



Some Famous HMM Tasks

Question 1: State Estimation

What is $P(q_T = \ldots | O_1 O_2 \ldots O_t)$



Some Famous HMM Tasks

Question 1: State Estimation

What is $P(q_T=S_i \mid O_1O_2\ldots O_t)$

Question 2: Most Probable Path

Given $O_1O_2\ldots O_T$, what is the most probable path that I took?

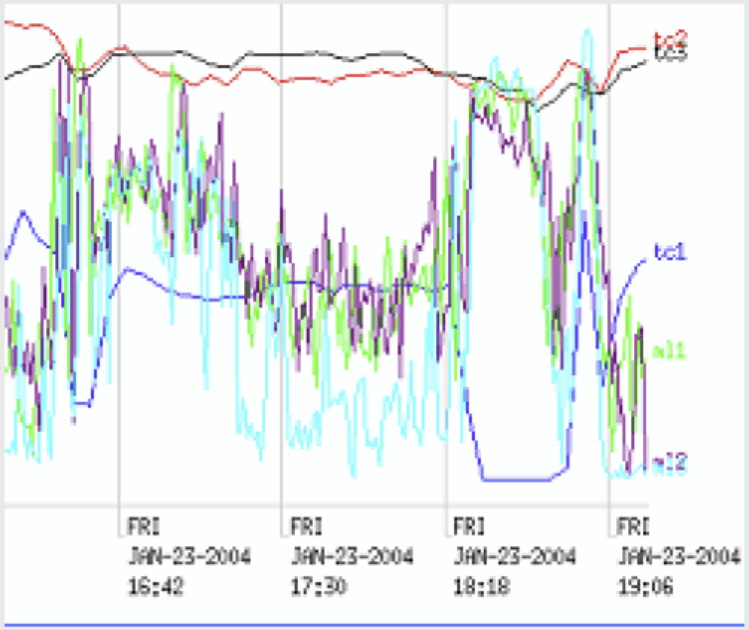
Some Famous HMM Tasks

Question 1: State Estimation

What is $P(q_T=S_i \mid O_1O_2\ldots O_t)$

Question 2: Most Probable Pat

Given $O_1O_2\ldots O_T$, what is the most probable path that I took?



Some Famous HMM Tasks

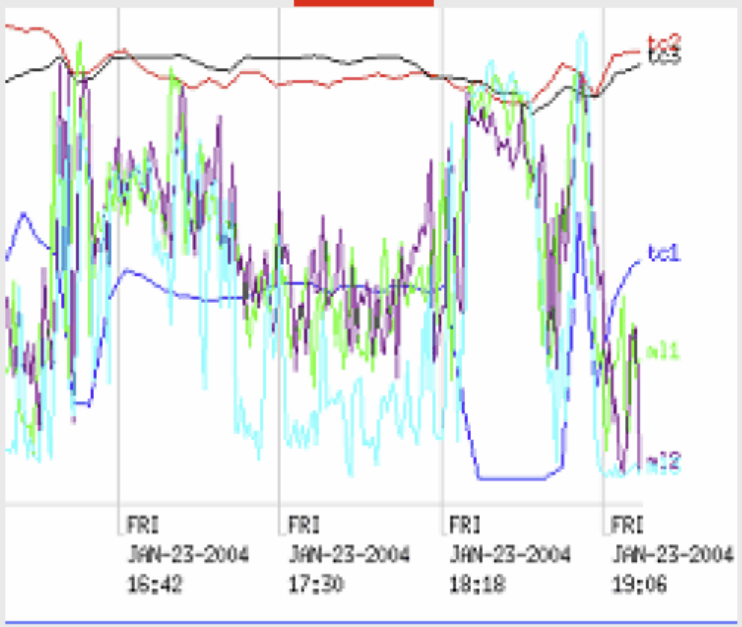
Question 1: State Estimation

What is $P(q_T=S_i \mid O_1O_2\ldots O_t)$

Question 2: Most Probable Pat

Given $O_1O_2\ldots O_T$, what is the most probable path that I took?

Woke up at 8.35, Got on Bus at 9.46, Sat in lecture 10.05-11.22…



Some Famous HMM Tasks

Question 1: State Estimation

What is $P(q_T=S_i \mid O_1O_2\ldots O_t)$

Question 2: Most Probable Path

Given $O_1O_2\ldots O_T$, what is the most probable path that I took?

Question 3: Learning HMMs:

Given $O_1O_2\ldots O_T$, what is the maximum likelihood HMM that could have produced this string of observations?

Some Famou

Question 1: State Estimation

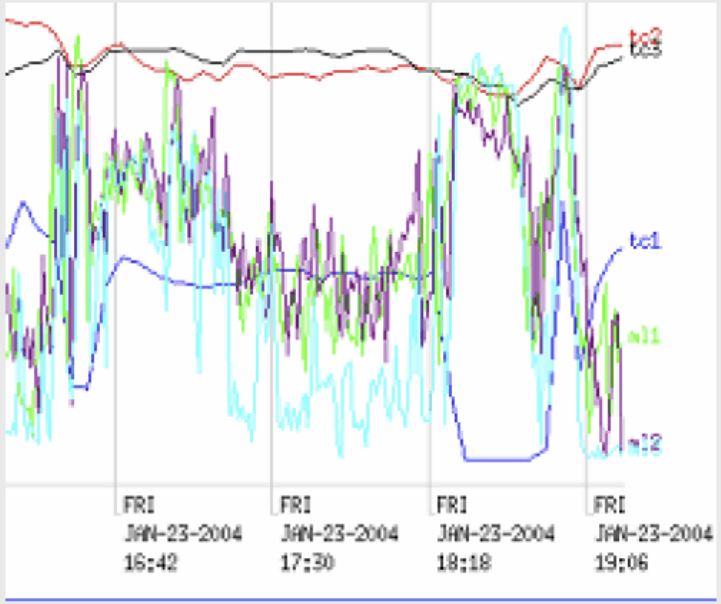
What is $P(q_T=S_i \mid O_1O_2\ldots O$

Question 2: Most Probable Pa

Given $O_1O_2\ldots O_T$, what is the most probable pat that I took?

Question 3: Learning HMMs:

Given $O_1O_2\ldots O_T$, what is the maximum likelihood HMM that could have produced this string of observations?



Some Famou

Question 1: State Estimation

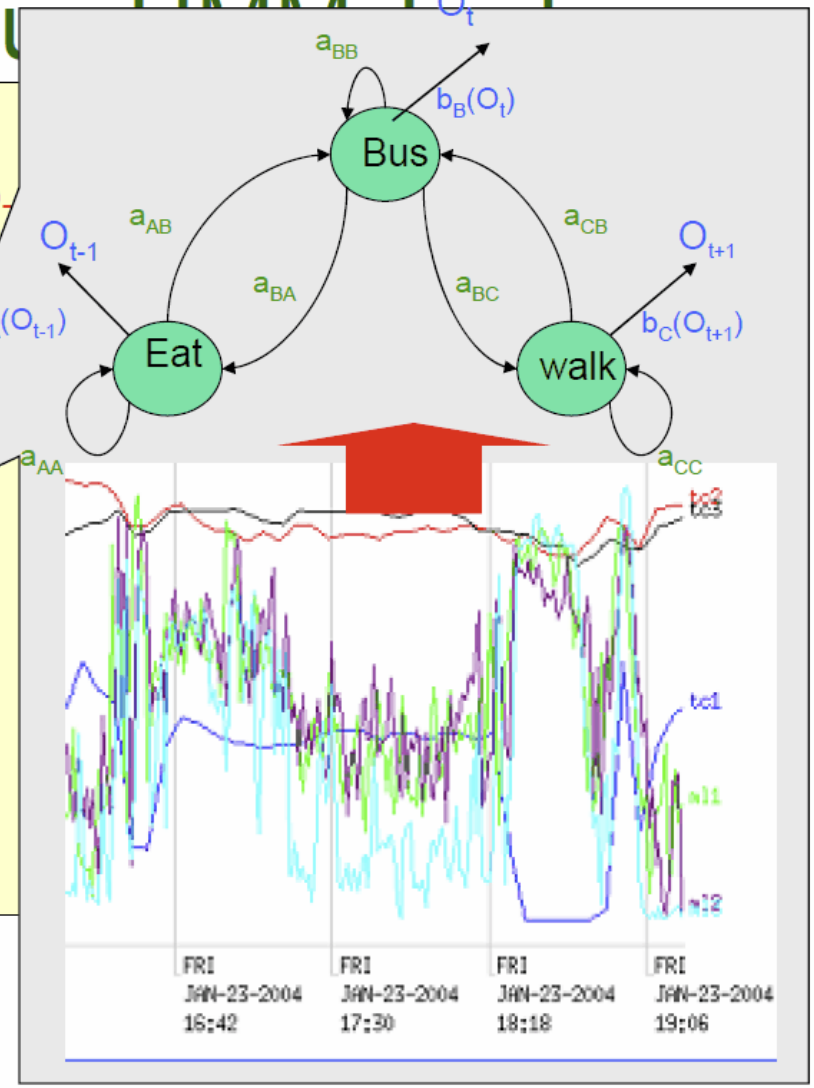
What is $P(q_T=S_i \mid O_1O_2\ldots O$-

Question 2: Most Probable Pa

Given $O_1O_2\ldots O_T$, what is the most probable pat that I took?

Question 3: Learning HMMs:

Given $O_1O_2\ldots O_T$, what is the maximum likelihood HMM that could have produced this string of observations?



Basic Operations in HMMs

For an observation sequence $O = O_1 \ldots O_T$, the three basic HMM operations are:

Problem	Algorithm	Complexity
Evaluation: Calculating $P(q_t=S_i \mid O_1 O_2 \ldots O_t)$	Forward-Backward	$O(TN^2)$
Inference: Computing $Q^* = argmax_Q P(Q\|O)$	Viterbi Decoding	$O(TN^2)$
Learning: Computing $\lambda^* = \arg max_\lambda P(O\|\lambda)$	Baum-Welch (EM)	$O(TN^2)$

T = # timesteps, N = # states

HMM Notation (from Rabiner's Survey)

*L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proc. of the IEEE, Vol.77, No.2, pp.257--286, 1989.

Available from

http://ieeexplore.ieee.org/iel5/5/698/00018626.pdf?arnumber=18626

The states are labeled $S_1\ S_2\ ..\ S_N$

For a particular trial….

Let T be the number of observations

T is also the number of states passed through

$O = O_1\ O_2\ ..\ O_T$ is the sequence of observations

$Q = q_1\ q_2\ ..\ q_T$ is the notation for a path of states

$\lambda = \langle N, M, \{\pi_i\}, \{a_{ij}\}, \{b_i(j)\}\rangle$ is the specification of an HMM

HMM Formal Definition

An HMM, λ, is a 5-tuple consisting of

- N the number of states
- M the number of possible observations
- $\{\pi_1, \pi_2, .. \pi_N\}$ The starting state probabilities

  $P(q_0 = S_i) = \pi_i$

This is new. In our previous example, start state was deterministic

- $$\left.\begin{matrix} a_{11} & a_{22} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & & \vdots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{matrix}\right\}$$

  The state transition probabilities

  $P(q_{t+1}=S_j \mid q_t=S_i)=a_{ij}$

- $$\left.\begin{matrix} b_1(1) & b_1(2) & \dots & b_1(M) \\ b_2(1) & b_2(2) & \dots & b_2(M) \\ \vdots & \vdots & & \vdots \\ b_N(1) & b_N(2) & \dots & b_N(M) \end{matrix}\right\}$$

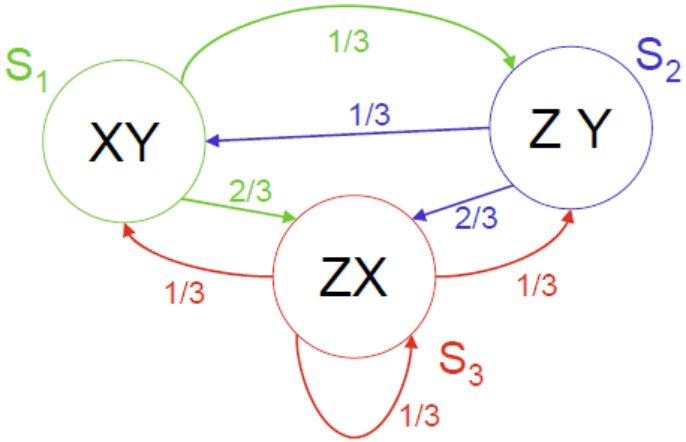
  The observation probabilities

  $P(O_t=k \mid q_t=S_i)=b_i(k)$

Here's an HMM

Start randomly in state 1 or 2

Choose one of the output symbols in each state at random.



N = 3

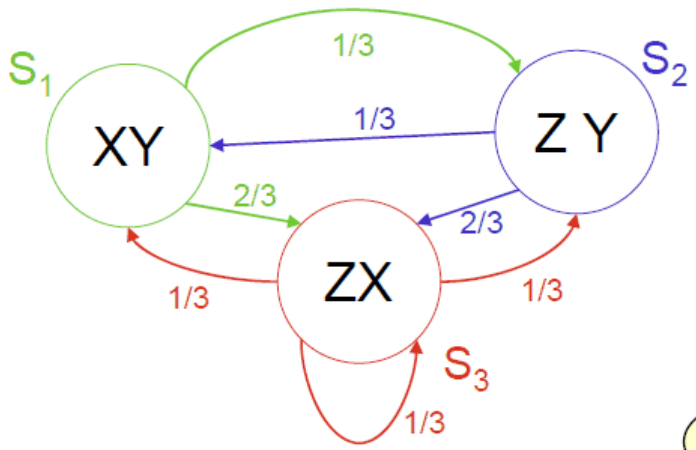
M = 3

$\pi_1 = 1/2$ $\pi_2 = 1/2$ $\pi_3 = 0$

$a_{11} = 0$	$a_{12} = 1/3$	$a_{13} = 2/3$
$a_{12} = 1/3$	$a_{22} = 0$	$a_{13} = 2/3$
$a_{13} = 1/3$	$a_{32} = 1/3$	$a_{13} = 1/3$

$b_1(X) = 1/2$	$b_1(Y) = 1/2$	$b_1(Z) = 0$
$b_2(X) = 0$	$b_2(Y) = 1/2$	$b_2(Z) = 1/2$
$b_3(X) = 1/2$	$b_3(Y) = 0$	$b_3(Z) = 1/2$

Here's an HMM



N = 3
M = 3

$\pi_1 = ½$ $\pi_2 = ½$ $\pi_3 = 0$

$a_{11} = 0$	$a_{12} = ⅓$	$a_{13} = ⅔$
$a_{12} = ⅓$	$a_{22} = 0$	$a_{13} = ⅔$
$a_{13} = ⅓$	$a_{32} = ⅓$	$a_{13} = ⅓$

b_1 (X) = ½	b_1 (Y) = ½	b_1 (Z) = 0
b_2 (X) = 0	b_2 (Y) = ½	b_2 (Z) = ½
b_3 (X) = ½	b_3 (Y) = 0	b_3 (Z) = ½

Start randomly in state 1 or 2

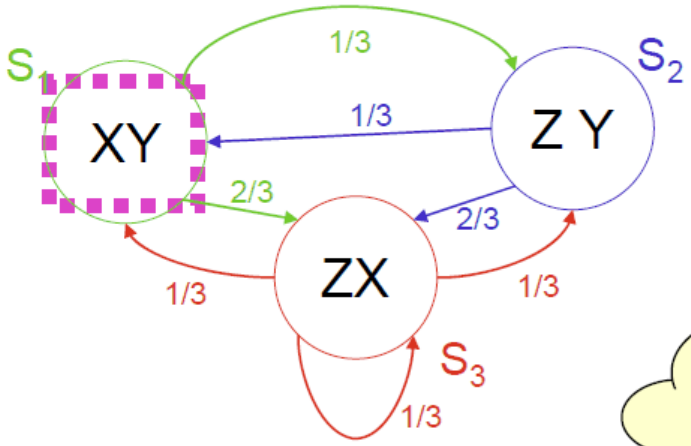
Choose one of the output symbols in each state at random.

Let's generate a sequence of observations:

50-50 choice between S_1 and S_2

q_0=	___	O_0=	___
q_1=	___	O_1=	___
q_2=	___	O_2=	___

Here's an HMM



Start randomly in state 1 or 2

Choose one of the output symbols in each state at random.

Let's generate a sequence of observations:

50-50 choice between X and Y

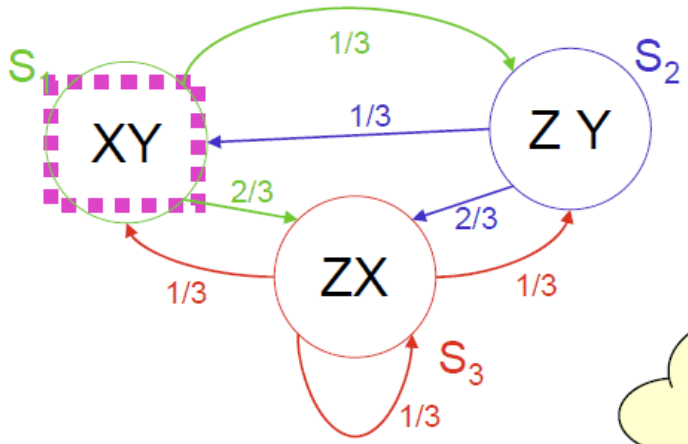
q_0=	S_1	O_0=	__		
q_1=	__	O_1=	__		
q_2=	__	O_2=	__		

N = 3

M = 3

$\pi_1 = ½$	$\pi_2 = ½$	$\pi_3 = 0$
$a_{11} = 0$	$a_{12} = ⅓$	$a_{13} = ⅔$
$a_{12} = ⅓$	$a_{22} = 0$	$a_{13} = ⅔$
$a_{13} = ⅓$	$a_{32} = ⅓$	$a_{13} = ⅓$
$b_1 (X) = ½$	$b_1 (Y) = ½$	$b_1 (Z) = 0$
$b_2 (X) = 0$	$b_2 (Y) = ½$	$b_2 (Z) = ½$
$b_3 (X) = ½$	$b_3 (Y) = 0$	$b_3 (Z) = ½$

Here's an HMM



Start randomly in state 1 or 2

Choose one of the output symbols in each state at random.

Let's generate a sequence of observations:

Goto S_3 with probability 2/3 or S_2 with prob. 1/3

N = 3

M = 3

$\pi_1 = ½$ $\pi_2 = ½$ $\pi_3 = 0$

$a_{11} = 0$ $a_{12} = ⅓$ $a_{13} = ⅔$

$a_{12} = ⅓$ $a_{22} = 0$ $a_{13} = ⅔$

$a_{13} = ⅓$ $a_{32} = ⅓$ $a_{13} = ⅓$

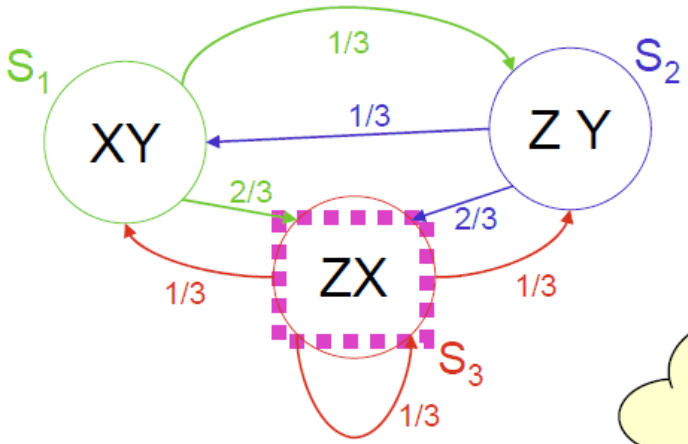
$b_1 (X) = ½$ $b_1 (Y) = ½$ $b_1 (Z) = 0$

$b_2 (X) = 0$ $b_2 (Y) = ½$ $b_2 (Z) = ½$

$b_3 (X) = ½$ $b_3 (Y) = 0$ $b_3 (Z) = ½$

q_0=	S_1	O_0=	X
q_1=	___	O_1=	___
q_2=	___	O_2=	___

Here's an HMM



N = 3
M = 3

$\pi_1 = ½$ $\pi_2 = ½$ $\pi_3 = 0$

$a_{11} = 0$	$a_{12} = ⅓$	$a_{13} = ⅔$
$a_{12} = ⅓$	$a_{22} = 0$	$a_{13} = ⅔$
$a_{13} = ⅓$	$a_{32} = ⅓$	$a_{13} = ⅓$

b_1 (X) = ½	b_1 (Y) = ½	b_1 (Z) = 0
b_2 (X) = 0	b_2 (Y) = ½	b_2 (Z) = ½
b_3 (X) = ½	b_3 (Y) = 0	b_3 (Z) = ½

Start randomly in state 1 or 2

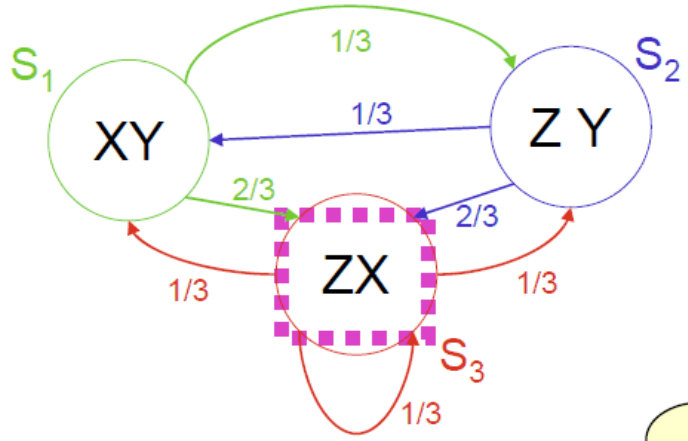
Choose one of the output symbols in each state at random.

Let's generate a sequence of observations:

50-50 choice between Z and X

q_0=	S_1	O_0=	X
q_1=	S_3	O_1=	___
q_2=	___	O_2=	___

Here's an HMM



Start randomly in state 1 or 2

Choose one of the output symbols in each state at random.

Let's generate a sequence of observations:

Each of the three next states is equally likely

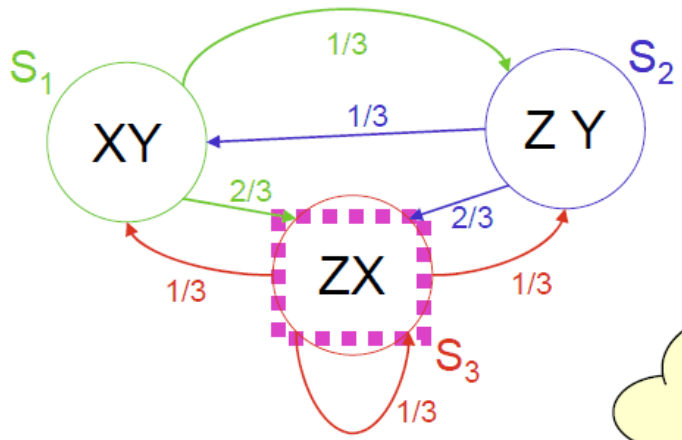
q_0=	S_1	O_0=	X
q_1=	S_3	O_1=	X
q_2=	___	O_2=	___

N = 3

M = 3

$\pi_1 = ½$	$\pi_2 = ½$	$\pi_3 = 0$
$a_{11} = 0$	$a_{12} = ⅓$	$a_{13} = ⅔$
$a_{12} = ⅓$	$a_{22} = 0$	$a_{13} = ⅔$
$a_{13} = ⅓$	$a_{32} = ⅓$	$a_{13} = ⅓$
$b_1 (X) = ½$	$b_1 (Y) = ½$	$b_1 (Z) = 0$
$b_2 (X) = 0$	$b_2 (Y) = ½$	$b_2 (Z) = ½$
$b_3 (X) = ½$	$b_3 (Y) = 0$	$b_3 (Z) = ½$

Here's an HMM



Start randomly in state 1 or 2

Choose one of the output symbols in each state at random.

Let's generate a sequence of observations:

50-50 choice between Z and X

N = 3

M = 3

$\pi_1 = ½$ $\pi_2 = ½$ $\pi_3 = 0$

$a_{11} = 0$ $a_{12} = ⅓$ $a_{13} = ⅔$

$a_{12} = ⅓$ $a_{22} = 0$ $a_{13} = ⅔$

$a_{13} = ⅓$ $a_{32} = ⅓$ $a_{13} = ⅓$

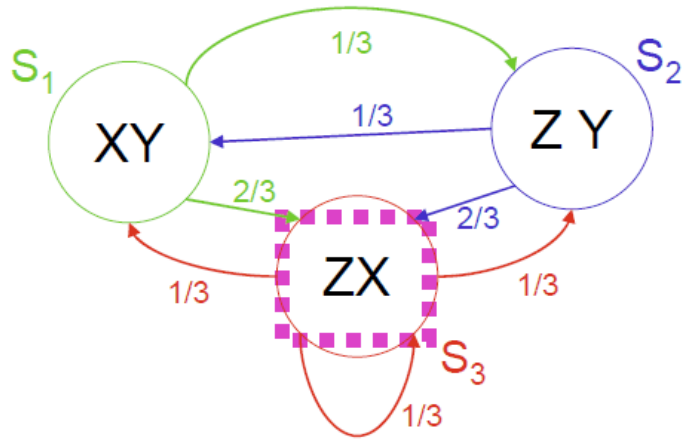
$b_1(X) = ½$ $b_1(Y) = ½$ $b_1(Z) = 0$

$b_2(X) = 0$ $b_2(Y) = ½$ $b_2(Z) = ½$

$b_3(X) = ½$ $b_3(Y) = 0$ $b_3(Z) = ½$

q_0=	S_1	O_0=	X
q_1=	S_3	O_1=	X
q_2=	S_3	O_2=	___

Here's an HMM



Start randomly in state 1 or 2

Choose one of the output symbols in each state at random.

Let's generate a sequence of observations:

N = 3

M = 3

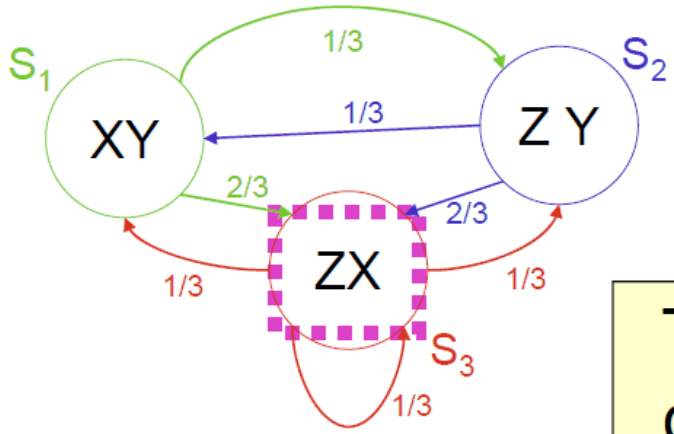
$\pi_1 = ½$ $\pi_2 = ½$ $\pi_3 = 0$

$a_{11} = 0$	$a_{12} = ⅓$	$a_{13} = ⅔$
$a_{12} = ⅓$	$a_{22} = 0$	$a_{13} = ⅔$
$a_{13} = ⅓$	$a_{32} = ⅓$	$a_{13} = ⅓$

b_1 (X) = ½	b_1 (Y) = ½	b_1 (Z) = 0
b_2 (X) = 0	b_2 (Y) = ½	b_2 (Z) = ½
b_3 (X) = ½	b_3 (Y) = 0	b_3 (Z) = ½

q_0=	S_1	O_0=	X
q_1=	S_3	O_1=	X
q_2=	S_3	O_2=	Z

State Estimation



Start randomly in state 1 or 2

Choose one of the output symbols in each state at random.

Let's generate a sequence of observations:

This is what the observer has to work with…

q_0=	?	O_0=	X
q_1=	?	O_1=	X
q_2=	?	O_2=	Z

$N = 3$

$M = 3$

$\pi_1 = ½$	$\pi_2 = ½$	$\pi_3 = 0$
$a_{11} = 0$	$a_{12} = ⅓$	$a_{13} = ⅔$
$a_{12} = ⅓$	$a_{22} = 0$	$a_{13} = ⅔$
$a_{13} = ⅓$	$a_{32} = ⅓$	$a_{13} = ⅓$
$b_1 (X) = ½$	$b_1 (Y) = ½$	$b_1 (Z) = 0$
$b_2 (X) = 0$	$b_2 (Y) = ½$	$b_2 (Z) = ½$
$b_3 (X) = ½$	$b_3 (Y) = 0$	$b_3 (Z) = ½$

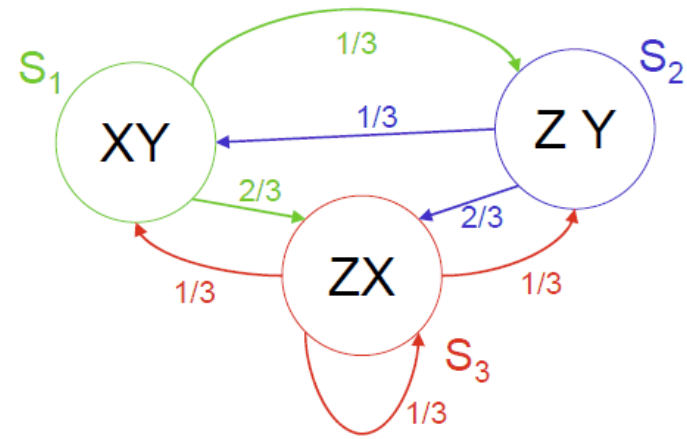
Prob. of a series of observations

What is $P(\mathbf{O}) = P(O_1\ O_2\ O_3) =$
$P(O_1 = X \wedge O_2 = X \wedge O_3 = Z)$?

Slow, stupid way:

$$P(\mathbf{O}) = \sum_{\mathbf{Q} \in \text{Paths of length 3}} P(\mathbf{O} \wedge \mathbf{Q})$$

$$= \sum_{\mathbf{Q} \in \text{Paths of length 3}} P(\mathbf{O} \mid \mathbf{Q}) P(\mathbf{Q})$$

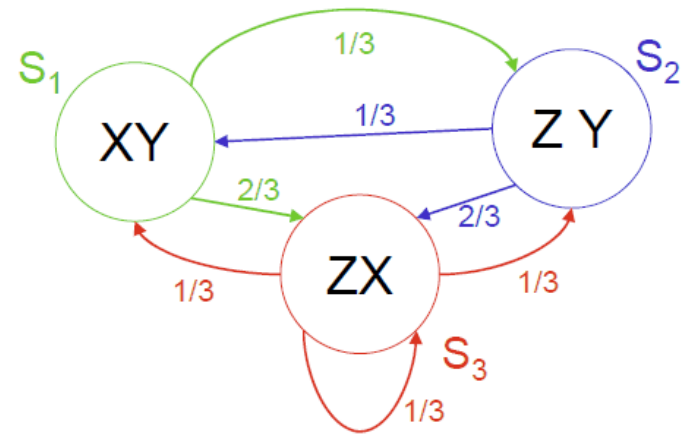


How do we compute P(Q) for an arbitrary path Q?

How do we compute P(O|Q) for an arbitrary path Q?

Prob. of a series of observations

What is $P(\mathbf{O}) = P(O_1\ O_2\ O_3) =$
$P(O_1 = X \wedge O_2 = X \wedge O_3 = Z)$?



Slow, stupid way:

$$P(\mathbf{O}) = \sum_{\mathbf{Q} \in \text{Paths of length 3}} P(\mathbf{O} \wedge \mathbf{Q})$$

$$= \sum_{\mathbf{Q} \in \text{Paths of length 3}} P(\mathbf{O} \mid \mathbf{Q}) P(\mathbf{Q})$$

How do we compute P(Q) for an arbitrary path Q?

How do we compute P(O|Q) for an arbitrary path Q?

$P(Q) = P(q_1, q_2, q_3)$

$= P(q_1)\ P(q_2, q_3 | q_1)$ (chain rule)

$= P(q_1)\ P(q_2 | q_1)\ P(q_3 | q_2, q_1)$ (chain)

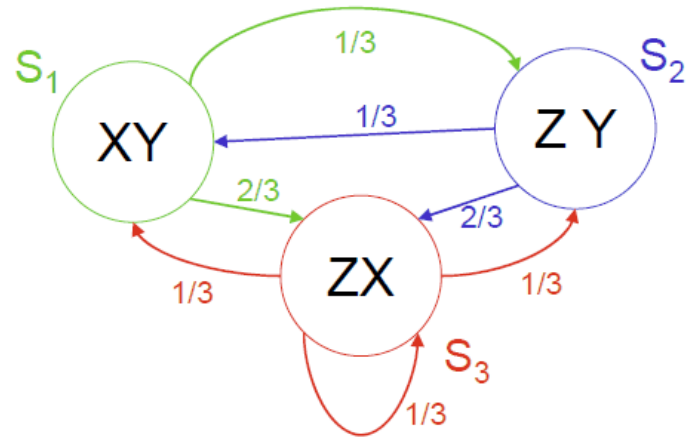
$= P(q_1)\ P(q_2 | q_1)\ P(q_3 | q_2)$ (why?)

Example in the case $Q = S_1\ S_3\ S_3$:

$= 1/2 * 2/3 * 1/3 = 1/9$

Prob. of a series of observations

What is $P(\mathbf{O}) = P(O_1\ O_2\ O_3) =$
$P(O_1 = X \wedge O_2 = X \wedge O_3 = Z)$?



Slow, stupid way:

$$P(\mathbf{O}) = \sum_{\mathbf{Q} \in \text{Paths of length 3}} P(\mathbf{O} \wedge \mathbf{Q})$$

$$= \sum_{\mathbf{Q} \in \text{Paths of length 3}} P(\mathbf{O} \mid \mathbf{Q}) P(\mathbf{Q})$$

How do we compute P(Q) for an arbitrary path Q?

How do we compute P(O|Q) for an arbitrary path Q?

P(O|Q)

$= P(O_1\ O_2\ O_3 \mid q_1\ q_2\ q_3)$

$= P(O_1 \mid q_1)\ P(O_2 \mid q_2)\ P(O_3 \mid q_3)$ (why?)

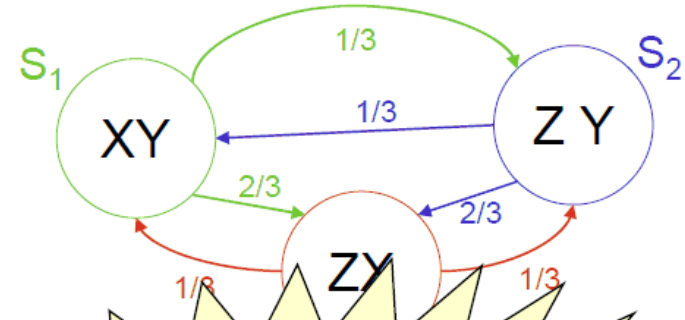
Example in the case $Q = S_1\ S_3\ S_3$:

$= P(X \mid S_1)\ P(X \mid S_3)\ P(Z \mid S_3) =$

$= 1/2 * 1/2 * 1/2 = 1/8$

Prob. of a series of observations

What is $P(\mathbf{O}) = P(O_1\ O_2\ O_3) =$
$P(O_1 = X \wedge O_2 = X \wedge O_3 = Z)$?



Slow, stupid way:

$$P(\mathbf{O}) = \sum_{\mathbf{Q} \in \text{Paths of length 3}} P(\mathbf{O} \wedge \mathbf{Q})$$

$$= \sum_{\mathbf{Q} \in \text{Paths of length 3}} P(\mathbf{O} \mid \mathbf{Q}) P(\mathbf{Q})$$

How do we compute P(Q) f
an arbitrary path Q?

How do we compute P(O|Q)
for an arbitrary path Q?

P(**O**) would need 27 P(Q) computations and 27 P(O|Q) computations

A sequence of 20 observations would need 3^{20} = 3.5 billion computations and 3.5 billion P(O|Q) computations

So let's be smarter…

The Prob. of a given series of observations, non-exponential-cost-style

Given observations $O_1\ O_2 \ldots O_T$

Define

$$\alpha_t(i) = P(O_1\ O_2 \ldots O_t \wedge q_t = S_i \mid \lambda) \qquad \text{where } 1 \le t \le T$$

$\alpha_t(i)$ = Probability that, in a random trial,

- We'd have seen the first t observations
- We'd have ended up in S_i as the t'th state visited.

In our example, what is $\alpha_2(3)$?

$\alpha_t(i)$: easy to define recursively

$\alpha_t(i) = P(O_1\ O_2 \ldots O_T \wedge q_t = S_i \mid \lambda)$ ($\alpha_t(i)$ can be defined stupidly by considering all paths length "t". How?)

$$\alpha_1(i) = P(O_1 \wedge q_1 = S_i)$$
$$= P(q_1 = S_i)P(O_1 | q_1 = S_i)$$
$$= \qquad \text{what?}$$
$$\alpha_{t+1}(j) = P(O_1 O_2 \ldots O_t O_{t+1} \wedge q_{t+1} = S_j)$$
$$=$$

$\alpha_t(i)$: easy to define recursively

$\alpha_t(i) = P(O_1\ O_2 \ldots O_T \wedge q_t = S_i \mid \lambda)$ ($\alpha_t(i)$ can be defined stupidly by considering all paths length "t". How?)

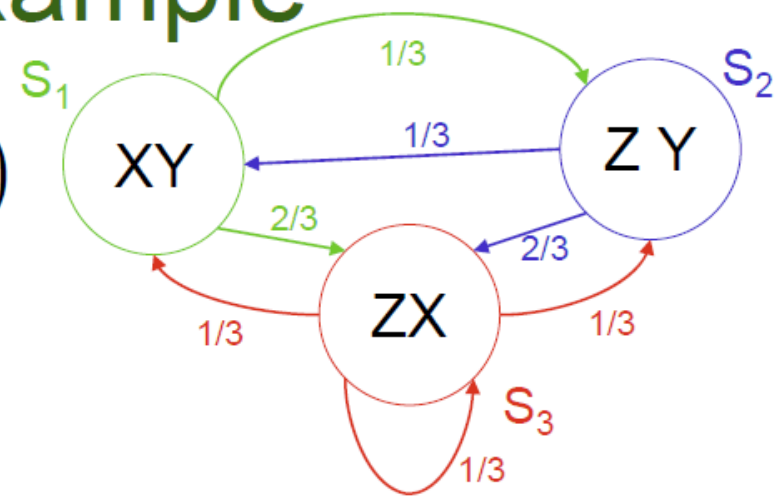
$$
\begin{aligned}
\alpha_1(i) &= P(O_1 \wedge q_1 = S_i) \\
&= P(q_1 = S_i) P(O_1 | q_1 = S_i) \\
&= \qquad\qquad\qquad\qquad \text{what?} \\
\alpha_{t+1}(j) &= P(O_1 O_2 \ldots O_t O_{t+1} \wedge q_{t+1} = S_j) \\
&= \sum_{i=1}^{N} P(O_1 O_2 \ldots O_t \wedge q_t = S_i \wedge O_{t+1} \wedge q_{t+1} = S_j) \\
&= \sum_{i=1}^{N} P(O_{t+1}, q_{t+1} = S_j | O_1 O_2 \ldots O_t \wedge q_t = S_i) P(O_1 O_2 \ldots O_t \wedge q_t = S_i) \\
&= \sum_{i} P(O_{t+1}, q_{t+1} = S_j | q_t = S_i) \alpha_t(i) \\
&= \sum_{i} P(q_{t+1} = S_j | q_t = S_i) P(O_{t+1} | q_{t+1} = S_j) \alpha_t(i) \\
&= \sum_{i} a_{ij} b_j(O_{t+1}) \alpha_t(i)
\end{aligned}
$$

in our example

$$\alpha_t(i) = \mathrm{P}(O_1 O_2 .. O_t \wedge q_t = S_i | \lambda)$$

$$\alpha_1(i) = b_i(O_1)\pi_i$$

$$\alpha_{t+1}(j) = \sum_i a_{ij} b_j(O_{t+1}) \alpha_t(i)$$



WE SAW $O_1\ O_2\ O_3$ = X X Z

$$\alpha_1(1) = \frac{1}{4} \qquad \alpha_1(2) = 0 \qquad \alpha_1(3) = 0$$

$$\alpha_2(1) = 0 \qquad \alpha_2(2) = 0 \qquad \alpha_2(3) = \frac{1}{12}$$

$$\alpha_3(1) = 0 \qquad \alpha_3(2) = \frac{1}{72} \qquad \alpha_3(3) = \frac{1}{72}$$

Easy Question

We can cheaply compute

$$\alpha_t(i)=P(O_1O_2\ldots O_t \wedge q_t=S_i)$$

(How) can we cheaply compute

$$P(O_1O_2\ldots O_t) \quad ?$$

(How) can we cheaply compute

$$P(q_t=S_i|O_1O_2\ldots O_t)$$

Easy Question

We can cheaply compute

$$\alpha_t(i)=P(O_1O_2\ldots O_t \wedge q_t=S_i)$$

(How) can we cheaply compute

$$P(O_1O_2\ldots O_t) \quad ?$$

$$\boxed{\sum_{i=1}^{N} \alpha_t(i)}$$

(How) can we cheaply compute

$$P(q_t=S_i|O_1O_2\ldots O_t)$$

$$\boxed{\frac{\alpha_t(i)}{\sum_{j=1}^{N} \alpha_t(j)}}$$

Most probable path given observations

What's most probable path given $O_1 O_2 \ldots O_T$, i.e.

What is $\underset{Q}{\operatorname{argmax}}\ \mathrm{P}(Q \mid O_1 O_2 \ldots O_T)$?

Slow, stupid answer:

$$\underset{Q}{\operatorname{argmax}}\ \mathrm{P}(Q \mid O_1 O_2 \ldots O_T)$$

$$= \underset{Q}{\operatorname{argmax}}\ \frac{\mathrm{P}(O_1 O_2 \ldots O_T \mid Q)\mathrm{P}(Q)}{\mathrm{P}(O_1 O_2 \ldots O_T)}$$

$$= \underset{Q}{\operatorname{argmax}}\ \mathrm{P}(O_1 O_2 \ldots O_T \mid Q)\mathrm{P}(Q)$$

Efficient MPP computation

We're going to compute the following variables:

$$\delta_t(i) = \max_{q_1 q_2 .. q_{t-1}} P(q_1\, q_2 \,..\, q_{t-1} \wedge q_t = S_i \wedge O_1 \,..\, O_t)$$

= The Probability of the path of Length t-1 with the maximum chance of doing all these things:

…OCCURING

and

…ENDING UP IN STATE S_i

and

…PRODUCING OUTPUT $O_1 \ldots O_t$

DEFINE: $mpp_t(i)$ = that path

So: $\delta_t(i) = \text{Prob}(mpp_t(i))$

The Viterbi Algorithm

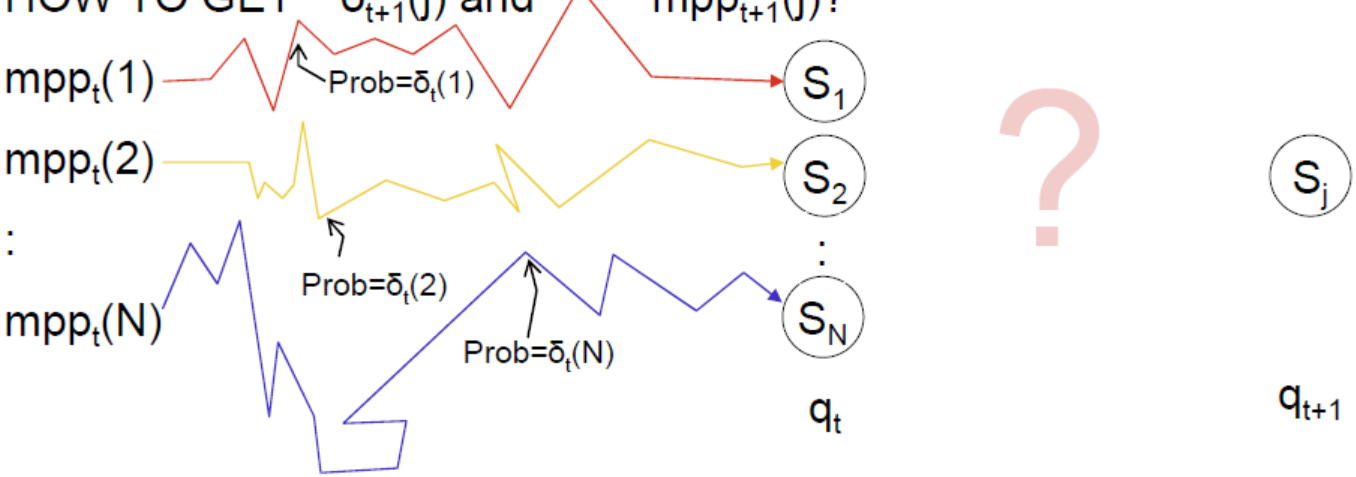
$$\delta_t(i) = \max_{q_1 q_2 \ldots q_{t-1}} \mathrm{P}(q_1 q_2 \ldots q_{t-1} \wedge q_t = S_i \wedge O_1 O_2 .. O_t)$$

$$mpp_t(i) = \arg\max_{q_1 q_2 \ldots q_{t-1}} \mathrm{P}(q_1 q_2 \ldots q_{t-1} \wedge q_t = S_i \wedge O_1 O_2 .. O_t)$$

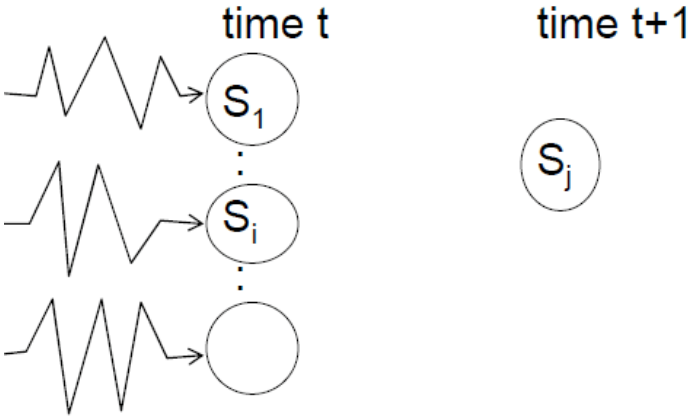
$$\begin{aligned}\delta_1(i) &= \max_{\text{one choice}} \mathrm{P}(q_1 = S_i \wedge O_1) \\ &= \mathrm{P}(q_1 = S_i)\mathrm{P}(O_1 | q_1 = S_i) \\ &= \pi_i b_i(O_1)\end{aligned}$$

Now, suppose we have all the $\delta_t(i)$'s and $mpp_t(i)$'s for all i.

HOW TO GET $\delta_{t+1}(j)$ and $mpp_{t+1}(j)$?



The Viterbi Algorithm

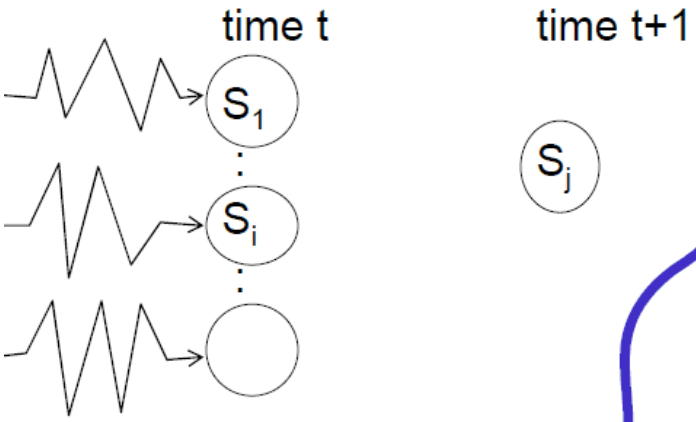


The most prob path with last two states S_i S_j

is

the most prob path to S_i , followed by transition $S_i \rightarrow S_j$

The Viterbi Algorithm



The most prob path with last two states S_i S_j

is

the most prob path to S_i , followed by transition $S_i \rightarrow S_j$

What is the prob of that path?

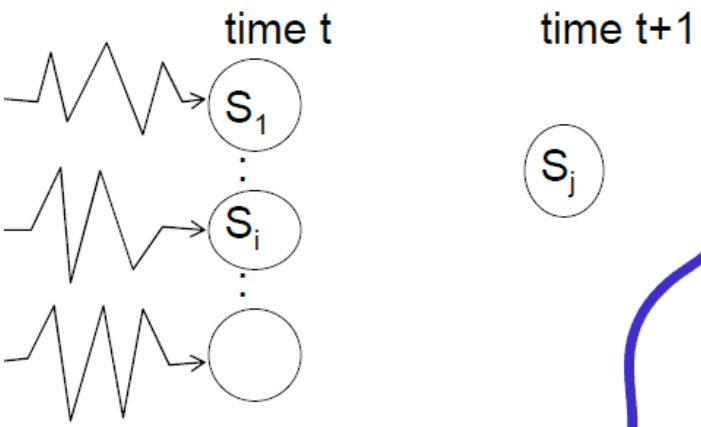
$$\delta_t(i) \times P(S_i \rightarrow S_j \wedge O_{t+1} \mid \lambda)$$

$$= \quad \delta_t(i)\, a_{ij}\, b_j\,(O_{t+1})$$

SO The most probable path to S_j has S_{i^*} as its penultimate state

where $i^* = \underset{i}{\operatorname{argmax}}\ \delta_t(i)\, a_{ij}\, b_j\,(O_{t+1})$

The Viterbi Algorithm



The most prob path with last two states S_i S_j

is

the most prob path to S_i , followed by transition $S_i \rightarrow S_j$

What is the prob of that path?

$$\delta_t(i) \times P(S_i \rightarrow S_j \wedge O_{t+1} \mid \lambda)$$

$$= \quad \delta_t(i)\, a_{ij}\, b_j\,(O_{t+1})$$

SO The most probable path to S_j has S_{i^*} as its penultimate state

where $i^* = \underset{i}{\operatorname{argmax}}\ \delta_t(i)\, a_{ij}\, b_j\,(O_{t+1})$

Summary:

$$\left.\begin{aligned} \delta_{t+1}(j) &= \delta_t(i^*)\, a_{ij}\, b_j\,(O_{t+1}) \\ mpp_{t+1}(j) &= mpp_{t+1}(i^*) S_{i^*} \end{aligned}\right\}$$ with i^* defined to the left

What's Viterbi used for?

Classic Example

Speech recognition:

Signal → words

HMM → observable is signal

→ Hidden state is part of word formation

What is the most probable word given this signal?

UTTERLY GROSS SIMPLIFICATION

In practice: many levels of inference; not one big jump.

HMMs are used and useful

But how do you design an HMM?

Occasionally, (e.g. in our robot example) it is reasonable to deduce the HMM from first principles.

But usually, especially in Speech or Genetics, it is better to infer it from large amounts of data. O_1 O_2 .. O_T with a big "T".

Observations previously in lecture → O_1 O_2 .. O_T

Observations in the next bit → O_1 O_2 .. $O_{\mathbf{T}}$

Inferring an HMM

Remember, we've been doing things like

$$P(O_1\ O_2\ ..\ O_T \mid \lambda\ )$$

That "λ" is the notation for our HMM parameters.

<u>Now</u> We have some observations and we want to estimate λ from them.

AS USUAL: We could use

(i) MAX LIKELIHOOD $\lambda = \underset{\lambda}{\operatorname{argmax}}\ P(O_1\ ..\ O_T \mid \lambda)$

(ii) BAYES

Work out $P(\ \lambda \mid O_1\ ..\ O_T\ )$

and then take $E[\lambda]$ or $\max_{\lambda} P(\ \lambda \mid O_1\ ..\ O_T\ )$

Max likelihood HMM estimation

Define

$\gamma_t(i) = P(q_t = S_i \mid O_1 O_2 \ldots O_T , \lambda)$

$\varepsilon_t(i,j) = P(q_t = S_i \wedge q_{t+1} = S_j \mid O_1 O_2 \ldots O_T , \lambda)$

$\gamma_t(i)$ and $\varepsilon_t(i,j)$ can be computed efficiently $\forall i,j,t$

(Details in Rabiner paper)

$$\sum_{t=1}^{T-1} \gamma_t(i) =$$ Expected number of transitions out of state i during the path

$$\sum_{t=1}^{T-1} \varepsilon_t(i,j) =$$ Expected number of transitions from state i to state j during the path

$$\gamma_t(i) = \mathrm{P}\left(q_t = S_i \middle| O_1 O_2 .. O_T, \lambda\right)$$

$$\varepsilon_t(i,j) = \mathrm{P}\left(q_t = S_i \wedge q_{t+1} = S_j \middle| O_1 O_2 .. O_T, \lambda\right)$$

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{expected number of transitions out of state i during path}$$

$$\sum_{t=1}^{T-1} \varepsilon_t(i,j) = \text{expected number of transitions out of i and into j during path}$$

HMM estimation

$$\text{Notice } \frac{\sum_{t=1}^{T-1} \varepsilon_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} = \frac{\begin{pmatrix}\text{expected frequency} \\ \mathrm{i} \to \mathrm{j}\end{pmatrix}}{\begin{pmatrix}\text{expected frequency} \\ \mathrm{i}\end{pmatrix}}$$

$$= \text{Estimate of Prob}\left(\text{Next state } \mathrm{S_j} \middle| \text{This state } \mathrm{S_i}\right)$$

We can re - estimate

$$\mathrm{a_{ij}} \leftarrow \frac{\sum \varepsilon_t(i,j)}{\sum \gamma_t(i)}$$

We can also re - estimate

$$\mathrm{b_j}(O_k) \leftarrow \cdots \qquad \text{(See Rabiner)}$$

We want a_{ij}^{new} = new estimate of $P(q_{t+1} = s_j \mid q_t = s_i)$

We want a_{ij}^{new} = new estimate of $P(q_{t+1} = s_j \mid q_t = s_i)$

$$= \frac{\text{Expected \# transitions } i \to j \mid \lambda^{old}, O_1, O_2, \cdots O_T}{\sum_{k=1}^{N} \text{Expected \# transitions } i \to k \mid \lambda^{old}, O_1, O_2, \cdots O_T}$$

We want a_{ij}^{new} = new estimate of $P(q_{t+1} = s_j \mid q_t = s_i)$

$$= \frac{\text{Expected \# transitions } i \to j \mid \lambda^{old}, O_1, O_2, \cdots O_T}{\sum_{k=1}^{N} \text{Expected \# transitions } i \to k \mid \lambda^{old}, O_1, O_2, \cdots O_T}$$

$$= \frac{\sum_{t=1}^{T} P(q_{t+1} = s_j, q_t = s_i \mid \lambda^{\text{old}}, O_1, O_2, \cdots O_T)}{\sum_{k=1}^{N} \sum_{t=1}^{T} P(q_{t+1} = s_k, q_t = s_i \mid \lambda^{\text{old}}, O_1, O_2, \cdots O_T)}$$

We want a_{ij}^{new} = new estimate of $P(q_{t+1} = s_j \mid q_t = s_i)$

$$= \frac{\text{Expected \# transitions } i \to j \mid \lambda^{old}, O_1, O_2, \cdots O_T}{\sum_{k=1}^{N} \text{Expected \# transitions } i \to k \mid \lambda^{old}, O_1, O_2, \cdots O_T}$$

$$= \frac{\sum_{t=1}^{T} P(q_{t+1} = s_j, q_t = s_i \mid \lambda^{\text{old}}, O_1, O_2, \cdots O_T)}{\sum_{k=1}^{N} \sum_{t=1}^{T} P(q_{t+1} = s_k, q_t = s_i \mid \lambda^{\text{old}}, O_1, O_2, \cdots O_T)}$$

$$= \frac{S_{ij}}{\sum_{k=1}^{N} S_{ik}} \text{ where } S_{ij} = \sum_{t=1}^{T} P(q_{t+1} = s_j, q_t = s_i, O_1, \cdots O_T \mid \lambda^{\text{old}})$$

$$= \text{What?}$$

We want a_{ij}^{new} = new estimate of $P(q_{t+1} = s_j \mid q_t = s_i)$

$$= \frac{\text{Expected \# transitions } i \to j \mid \lambda^{old}, O_1, O_2, \cdots O_T}{\sum_{k=1}^{N} \text{Expected \# transitions } i \to k \mid \lambda^{old}, O_1, O_2, \cdots O_T}$$

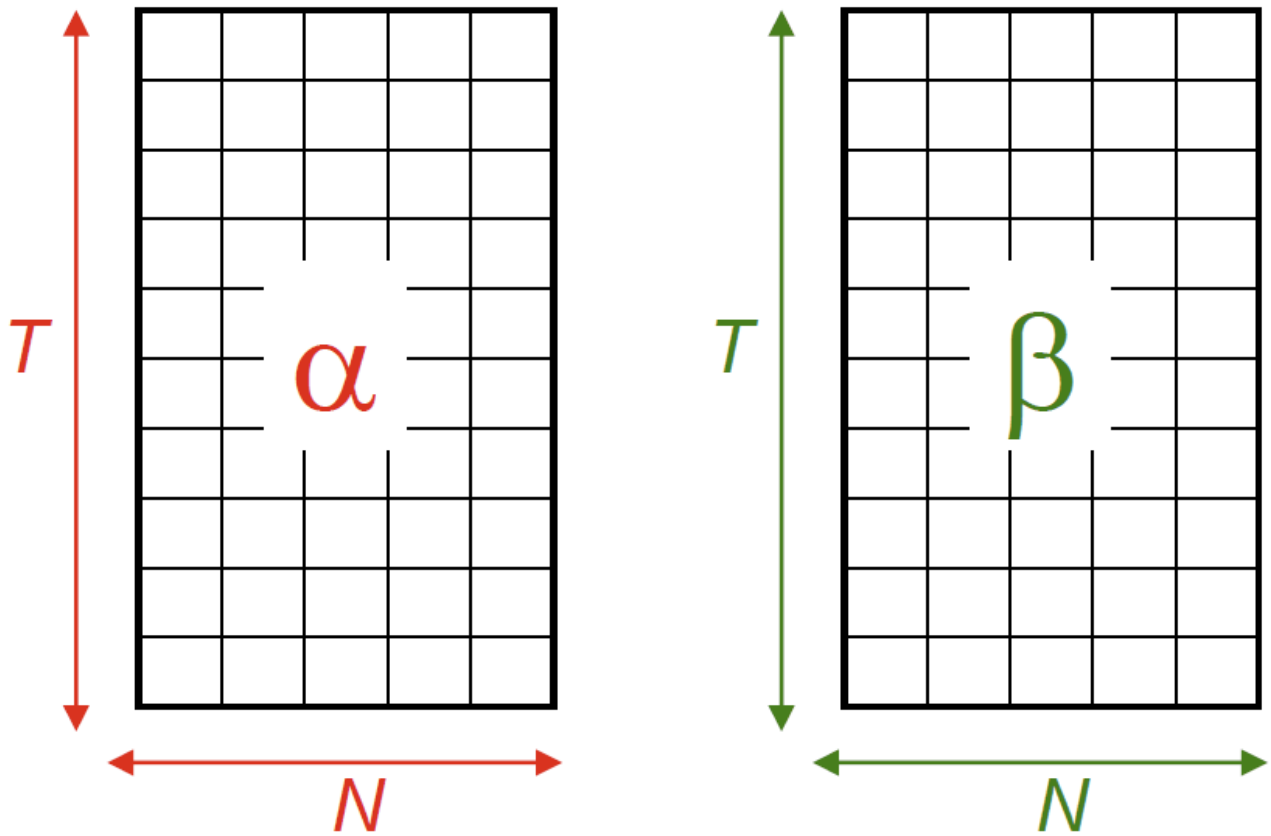
$$= \frac{\sum_{t=1}^{T} P(q_{t+1} = s_j, q_t = s_i \mid \lambda^{\text{old}}, O_1, O_2, \cdots O_T)}{\sum_{k=1}^{N} \sum_{t=1}^{T} P(q_{t+1} = s_k, q_t = s_i \mid \lambda^{\text{old}}, O_1, O_2, \cdots O_T)}$$

$$= \frac{S_{ij}}{\sum_{k=1}^{N} S_{ik}} \quad \text{where} \quad S_{ij} = \sum_{t=1}^{T} P(q_{t+1} = s_j, q_t = s_i, O_1, \cdots O_T \mid \lambda^{\text{old}})$$

$$= a_{ij} \sum_{t=1}^{T} \alpha_t(i) \beta_{t+1}(j) b_j(O_{t+1})$$

We want $a_{ij}^{\text{new}} = S_{ij} \Big/ \sum_{k=1}^{N} S_{ik}$ where $S_{ij} = a_{ij} \sum_{t=1}^{T} \alpha_t(i)\beta_{t+1}(j)b_j(O_{t+1})$

We want $a_{ij}^{\text{new}} = S_{ij} \Big/ \sum_{k=1}^{N} S_{ik}$ where $S_{ij} = a_{ij} \sum_{t=1}^{T} \alpha_t(i)\beta_{t+1}(j)b_j(O_{t+1})$



EM for HMMs

If we knew λ we could estimate EXPECTATIONS of quantities such as

Expected number of times in state i

Expected number of transitions $i \rightarrow j$

If we knew the quantities such as

Expected number of times in state i

Expected number of transitions $i \rightarrow j$

We could compute the MAX LIKELIHOOD estimate of

$$\lambda = \langle \{a_{ij}\}, \{b_i(j)\}, \pi_i \rangle$$

Roll on the EM Algorithm…

EM 4 HMMs

1. Get your observations $O_1 \ldots O_T$
2. Guess your first λ estimate $\lambda(0)$, $k=0$
3. $k = k+1$
4. Given $O_1 \ldots O_T$, $\lambda(k)$ compute
   $\gamma_t(i)$, $\varepsilon_t(i,j)$ $\quad \forall 1 \le t \le T, \quad \forall 1 \le i \le N, \quad \forall 1 \le j \le N$
5. Compute expected freq. of state i, and expected freq. $i \rightarrow j$
6. Compute new estimates of a_{ij}, $b_j(k)$, π_i accordingly. Call them $\lambda(k+1)$
7. Goto 3, unless converged.

- **Also known (for the HMM case) as the BAUM-WELCH algorithm.**

Bad News

- There are lots of local minima

Good News

- The local minima are usually adequate models of the data.

Notice

- EM does not estimate the number of states. That must be given.
- Often, HMMs are forced to have some links with zero probability. This is done by setting $a_{ij}=0$ in initial estimate $\lambda(0)$
- Easy extension of everything seen today: HMMs with real valued outputs

Bad News

- There are lots o

- The local minim
data.

Trade-off between too few states (inadequately modeling the structure in the data) and too many (fitting the noise).

Thus #states is a regularization parameter.

Blah blah blah… bias variance tradeoff…blah blah…cross-validation…blah blah….AIC, BIC….blah blah (same ol' same ol')

Notice

- EM does not estimate the number of states. That must be given.
- Often, HMMs are forced to have some links with zero probability. This is done by setting $a_{ij}=0$ in initial estimate $\lambda(0)$
- Easy extension of everything seen today: HMMs with real valued outputs

What You Should Know

- What is an HMM ?
- Computing (and defining) $\alpha_t(i)$
- The Viterbi algorithm
- Outline of the EM algorithm
- To be very happy with the kind of maths and analysis needed for HMMs
- Fairly thorough reading of Rabiner* up to page 266* [Up to but not including "IV. Types of HMMs"].

DON'T PANIC: starts on p. 257.

*L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proc. of the IEEE, Vol.77, No.2, pp.257--286, 1989.

http://ieeexplore.ieee.org/iel5/5/698/00018626.pdf?arnumber=18626

And now…

Applications in Bioinformatics

Segmentation of sequences

- Switching between fair and loaded dice

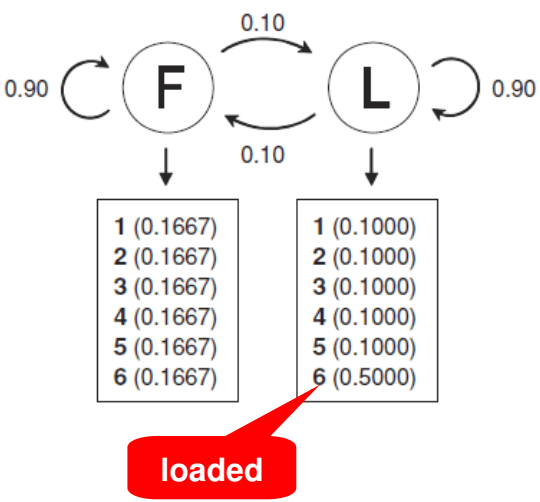
An example of visible sequence:

S = `4553653163363555133362665132141636651666`

If we know the properties of the two dice and of the underlying HMM, can be find the most likely sequence of hidden states behind it? i.e. can we guess which die was used at each time point in the sequence?

Segmentation of sequences

- Switching between fair and loaded dice



An example of visible sequence:

O = `4553653163363555133362665132141636651666`

If we know the properties of the two dice and of the underlying HMM, can be find the most likely sequence of hidden states behind it? i.e. can we guess which die was used at each time point in the sequence?

Visible *O* = `4553653163363555133362665132141636651666`

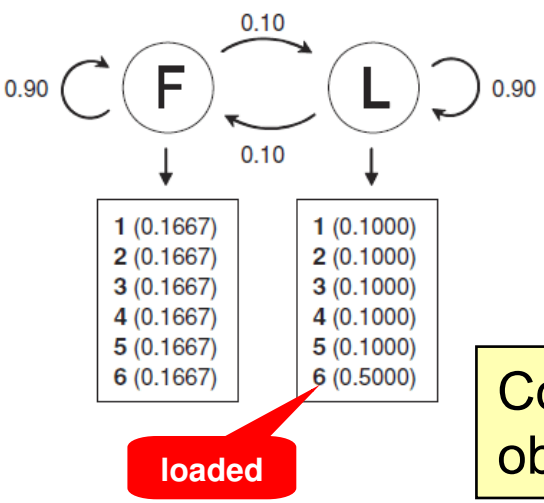
Segmentation of sequences

- Switching between fair and loaded dice

An example of visible sequence:

O = `4553653163363555133362665132141636651666`

If we know the properties of the two dice and of the underlying HMM, can be find the most likely sequence of hidden states behind it? i.e. can we guess which die was used at each time point in the sequence?



Visible *O* = `4553653163363555133362665132141636651666`

Hidden *Q* = `1111111111111111111122222111111222222222`

Computation: Viterbi Algorithm, given a sequence of observations, what is the most probable path that I took

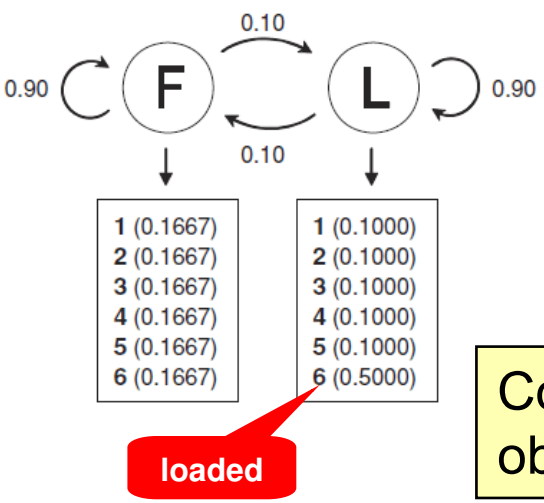
Segmentation of sequences

- Switching between fair and loaded dice

An example of visible sequence:

O = `4553653163363555133362665132141636651666`

If we know the properties of the two dice and of the underlying HMM, can be find the most likely sequence of hidden states behind it? i.e. can we guess which die was used at each time point in the sequence?



Visible *O* = `4553653163363555133362665132141636651666`

Hidden *Q* = `1111111111111111111122222111111222222222`

Computation: Viterbi Algorithm, given a sequence of observations, what is the most probable path that I took

Segmentation = detecting boundaries between statistically different regions.

But we can also estimate the model parameters given some training data where both the hidden and the observed states are known → EM algorithm

The anatomy of a genome (1)

- Genome = set of all DNA contained in a cell.
- Formed by one or more long stretches of DNA strung together into *chromosomes*.
- Chromosomes are faithfully replicated by a cell when it divides.

- The set of chromosomes in a cell contains the DNA necessary to synthesize the *proteins* and other molecules needed to survive, as well as much of the information necessary to finely regulate their synthesis
  - Each protein is coded for by a specific *gene*, a stretch of DNA containing the information necessary for that purpose.

The anatomy of a genome (2)

- DNA molecules consist of a chain of smaller molecules called *nucleotides* that are distinct from each other only in a chemical element called a *base*.

- For biochemical reasons, DNA sequences have an orientation
  - It is possible to distinguish a specific direction in which to read each chromosome or gene
  - The directions are often represented as the left and right end of the sequence

- A DNA sequence can be single-stranded or double-stranded.
- The double-stranded nature is caused by the *pairing* of bases (base pairs, bp).
- When it is double-stranded, the two strands have opposite direction and are complementary to one another.
- This complementarity means that for each A, C, G, T in one strand, there is a T, G, C, or A, respectively, in the other strand.

The anatomy of a genome (3)

- Chromosomes are double-stranded (➔"double helix")
- Information about a gene can be contained in either strand.
- This pairing introduces a complete redundancy in the encoding
  - allows the cell to reconstitute the entire genome from just one strand (enables faithful replication)
  - for simple convenience, we usually just write out the single strand of DNA sequence we are interested in from left to right

- The letters of the DNA alphabet are variously called nucleotides (nt), bases, or base pairs (bp) for double stranded DNA.
- The length of a DNA sequence can be measured in bases, or in kilobases (1000 bp or Kb) or megabases (1000000 bp or Mb).
- The genomes present in different organisms range in size from kilobases to megabases.

Viral genomes

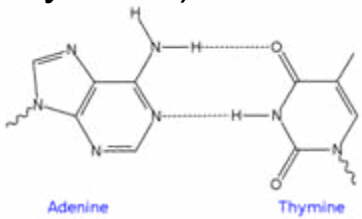
- At least 1000 viral genomes have been sequenced (2006 data), starting from what is considered the “pre-genomic” era (late 1970s).
- They are usually very short (5 to 50 Kb) and contain very few genes.
- Their sequencing was a milestone for biology.
- They enabled scientists to develop conceptual tools that would become essential for the analysis of the genomes of larger, free-living organisms.
- Their analysis is also highly relevant for epidemiological and clinical applications, as has been demonstrated in cases involving HIV and SARS.
- Peculiarly, viral genomes can be either single or double-stranded, and either DNA- or RNA-based.
- Because of their small size, we can analyze a large number of viral genomes simultaneously on a laptop, a task that would require a large cluster of machines in the case of longer genomic sequences.

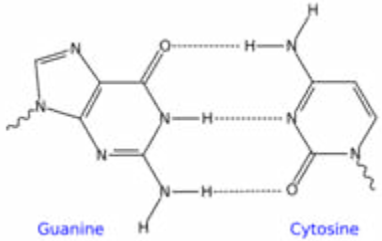
The λ-phage virus genome

- Phages are viruses that infect bacteria, and *λ-phage* infects the bacterium *E. coli*, a very well-studied model system.
- *Bacteriophage λ* was one of the first viral genomes completely sequenced (1982). It is 48502 bases long.

Organism	Completion date	Size	Description
phage phiX174	1978	5,368 bp	1st viral genome
human mtDNA	1980	16,571 bp	1st organelle genome
lambda phage	1982	48,502 bp	important virus model
HIV	1985	9,193 bp	AIDS retrovirus
H. influenzae	1995	1,830 Kb	1st bacterial genome
M. genitalium	1995	580 Kb	smallest bacterial genome
S. cerevisiae	1996	12.5 Mb	1st eukaryotic genome
E. coli K12	1997	4.6 Mb	bacterial model organism
C. trachomatis	1998	1,042 Kb	internal parasite of eukaryotes
D. melanogaster	2000	180 Mb	fruit fly, model insect
A. thaliana	2000	125 Mb	thale cress, model plant
H. sapiens	2001	3,000 Mb	human
SARS	2003	29,751 bp	coronavirus

bp (base pair) = two nucleotides on opposite complementary DNA or RNA strands connected via hydrogen bonds (in DNA, adenine forms a base pair with thymine, as does guanine with cytosine).





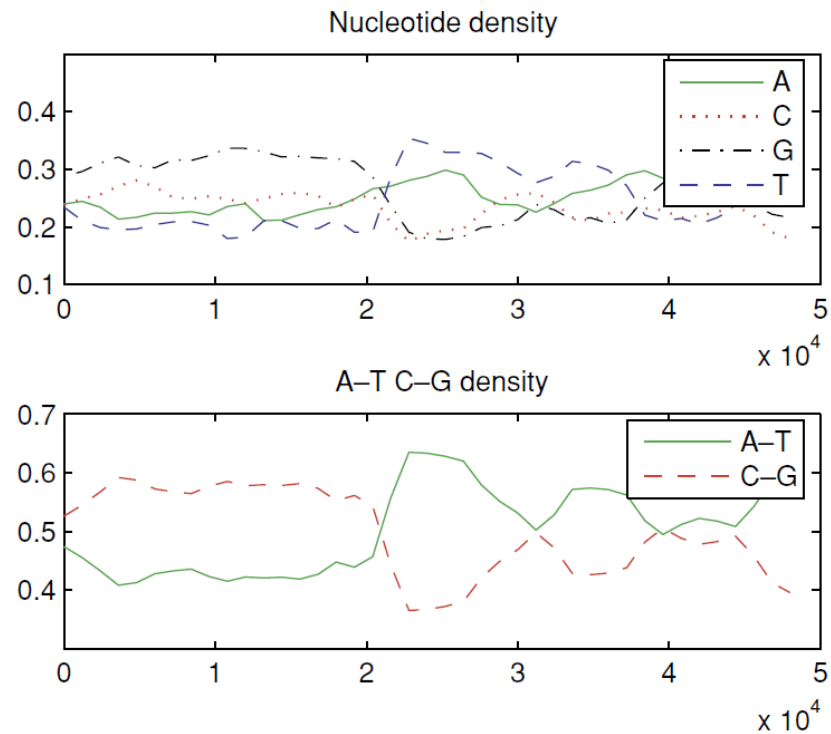
Example of an 18 base-paired DNA sequence:

ATCGATTGAGCTCTAGCG

TAGCTAACTCGAGATCGC

Change point analysis and the λ-phage

- The analysis of frequencies of the 4 nucleotides is overly complex for most biological needs.
- What most papers report (and is all that is generally necessary) is the aggregate frequencies for C and G (called GC content) versus the aggregate frequencies for A and T (AT content).
- Given that these two quantities are required to always sum to 1, only the GC content is typically reported.
- The motivation for reporting simply the GC content is that –due to a number of chemical reasons– the content of G and C in a genome is often very similar, as is the content of A and T.
- In this way, only one value needs to be reported instead of four.

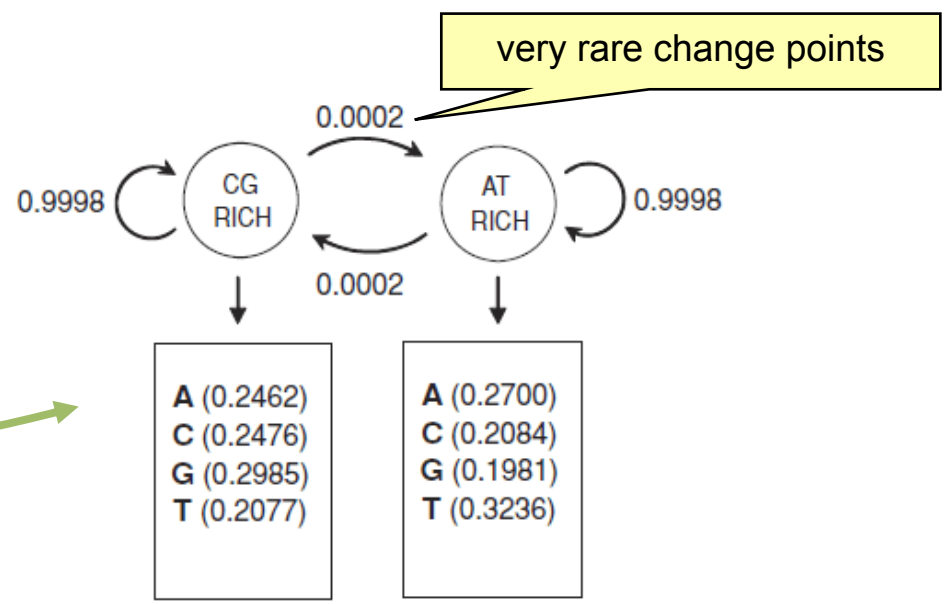


Nucleotide density in λ-phage

- The phage genome is composed of two halves with completely different GC content: the first half G+C rich, the second A+T rich.
  This is a simple example of a change point in a genome, clearly dividing it into homogeneous regions of base composition.

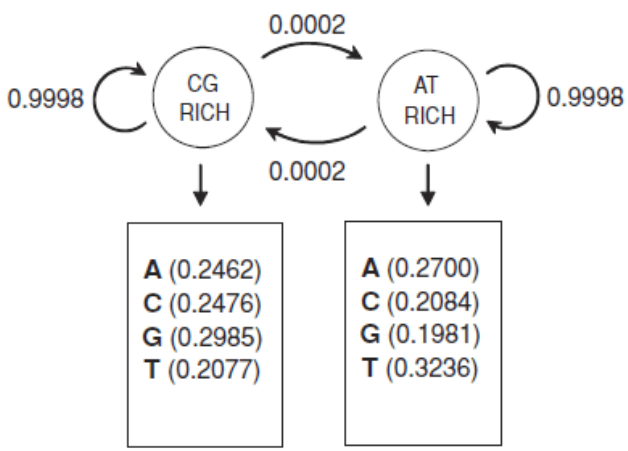
Segmentation of the λ-phage genome (1…)

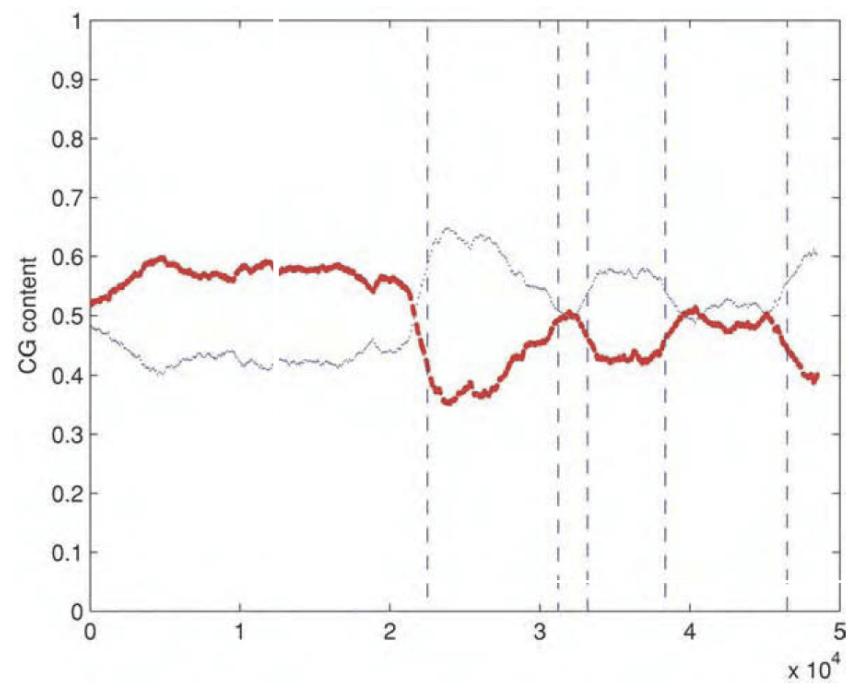
- Use HMM to segment the λ-phage genome into blocks of GC-rich subsequences and AT-rich subsequences.
- Phase 1: learning HMM
  - Start with random transition (a) and emission (b) matrices for HMM.
  - Use EM algorithm to better estimate those parameters (assuming 2 hidden states and 4 observable symbols).



Segmentation of the λ-phage genome (and 2)

- Phase 2: inference with the HMM
  - Use Viterbi algorithm to get the segmentation of the GC content plot.





segmentation found by a two-state HMM

Sequence alignment

- It is probably the most important task in bioinformatics. Many uses:
  - Prediction of function
  - Database searching
  - Gene finding
  - Sequence divergence
  - Sequence assembly
- It is routinely applied to both amino acid and DNA sequences.
- Its ultimate purpose is to measure sequence similarity, or how closely sequences resemble each other.

Pairwise sequence alignment

- *Global* alignment of two sequences (a.k.a. *pairwise* alignment)
  - It is a representation of the correspondence between their respective symbols (i.e. their nucleotides).
  - If two sequences have the same ancestor, we expect them to have many symbols –and indeed entire substrings– in common.

```
V I V A L A S V E G A S
| | | |   |   |       |
V I V A D A - V - - I S
```

  - To identify the corresponding homologous position in the other sequence.
  - Mutations between the sequences appear as mismatches and *indels* (**in**sertions or **del**etions) appear as gaps in one of the two sequences.
  - Because we do not know what the ancestor of these two sequences looked like, we do not know if the length difference is due to insertions in one sequence, deletions in the other, or some combination of the two.

Optimal global alignment

- *Scoring function* of a pair of symbols in position *i* of the alignment: $\sigma(x_i, y_i)$
  - Example

    $\sigma(-, a) = \sigma(a, -) = \sigma(a, b) = -1 \quad \forall a \neq b$

    $\sigma(a, b) = 1 \quad \forall a = b$

$$\begin{array}{c} \ldots\ x_i\ \ldots \\ | \\ \ldots\ y_i\ \ldots \end{array}$$

- *Total alignment score:*

$$M = \sum_{i=1}^{c} \sigma(x_i, y_i)$$

- *Optimal* global alignment of strings **s** and **t**:
  - the alignment of **s** and **t** that maximizes the total alignment score over all possible alignments

Local alignment

- More realistic situation: we are interested in the best alignment between two *parts* of **s** and **t** (that is, two subsequences)
  - two homologous regions of DNA might contain smaller conserved elements within them
- The best alignment of subsequences of **s** and **t** is called the *optimal local alignment*
- This can be thought of as removing a prefix and a suffix in each of the two sequences, and testing how well we can align the remaining internal substrings.

Multiple alignment of sequences

- Problem in computational genomics:
  - To characterize **sets of** homologous proteins (gene families) based on common patterns in their sequence.
  - This allows us, for example, to determine if a new protein belongs to a certain family or not.
- We introduce a *"profile HMM"* (pHMM):
  - pHMMs can be seen as abstract descriptions of a protein family, or statistical summaries of a multiple sequence alignment.
  - They are constructed from multiple alignments of homologous sequences.
  - They contain *match* states, which describe the distribution of amino acids at each position, as well as *insertion* and *deletion* states that allow for the addition or removal of residues.
  - There is a *match* state, *insertion* state, and *deletion* state for each column of a multiple alignment
  - For each match and insertion state there is a specific probability of emitting each of the 20 amino acids. No amino acids are emitted from deletion states.

Profile HMMs for multiple alignment

they start with the same 4 amino acids

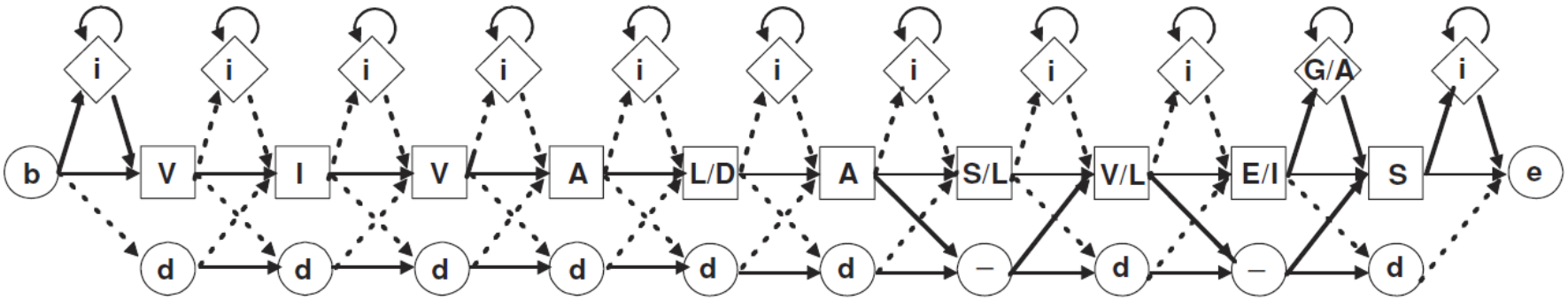
```
V I V A L A S V E G A S
V I V A D A - V I - - S
V I V A D A L L - - A S
```

all finish with symbol S

then various choices are possible

common amino acid

variable # of positions with ≠ amino acids



- HMM: each path between beginning and end nodes represents a possible sequence
- Transitions with low probability are denoted by dotted lines, and those with high probability by solid lines
- At each square node, a symbol can be emitted, according to the emission probability associated with that position. For readability, we write only the dominant symbols of the emission matrix (in general any symbol is possible, with different probabilities)
- Insertion (diamonds) and deletion (circles) states are present, so certain paths allow us to insert gaps or extra symbols in the profile
- **This model allows to compute the degree to which a given sequence fits the model**

Profile HMMs for multiple alignment

- Profile HMMs allow us to summarize the salient features of a protein alignment into a single model, against which novel sequences can easily be tested for similarity.
- Also, since pHMMs are an abstract representation of a multiple alignment, they can be used to *produce* pairwise or multiple alignments; sequences are said to be aligned to the model.
- Aligning a sequence with a pHMM is equivalent to aligning it with the hundreds of sequences used to produce the model.
- There are free online repositories, like Pfam, that store pHMMs of many protein families.

Case study: odorant receptors

- What you should be able to do:
  - Read and understand section 4.5 of the book (*) *Introduction to Computational Genomics: A Case Studies Approach*, by Cristianini and Hahn.
    - To see HMMs in action by studying the protein family to which odorant receptors (ORs) belong:

      7-transmembrane (7-TM) G-protein coupled receptors
    - This is an important family containing (in humans) 250 proteins in addition to the 400 ORs.
      - It includes receptors found in the retina to sense light as well as receptors for hormones and neurotransmitters such as melatonin, serotonin, and dopamine.
      - More than half of today's pharmaceuticals target these receptors.

(*) Also available at the course webpage ("material adicional").

Bibliography

- L. R. Rabiner: "A tutorial on Hidden Markov Models and selected applications in speech recognition". *Proceedings of the IEEE*, vol. 77, no.2, pp. 257–286, February 1989.

- R. Durbin, S.R. Eddy, A. Krogh, and G. Mitchison: *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1999.

- N. Cristianini and M.W. Hahn: *Introduction to Computational Genomics: A Case Studies Approach*. Cambridge University Press, 2006.