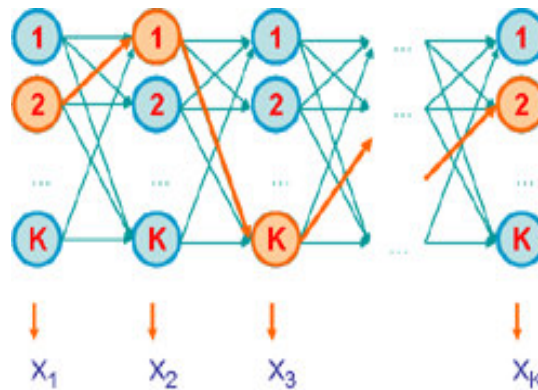


**CS262 Computational Genomics**  
**Lecture 6: Hidden Markov Models (Contd.)**  
**Professor Serafim Batzoglou**

**Topics covered in this lecture are:**

- 1 Review of algorithms for decoding and evaluation problems of HMM
- 2 A Biological example where HMM is applicable(CpG Islands)
- 3 The “LEARNING” problem in HMM
- 4 Variants of HMM



**1. Review of Algorithms for decoding and evaluation problems of HMM**

In the last lecture we discussed what a Hidden Markov Model (HMM) is and what are the kind of problems that one might be interested when modeling a given real world problem as an HMM.

We discussed the following algorithms that are used to solve the evaluation and decoding problem of HMMs.

**i. Viterbi Algorithm:** This algorithm is used to find the most likely parse for a given sequence, given an HMM along with its associated parameters. To recall, once again, a parse is a sequence of states of the HMM such that at state  $s_i$  the letter  $x_i$  of the sequence was emitted.

e.g.: In the casino player example, finding the loaded and fair die roll regions or finding coding and non-coding regions in a gene.

**ii. Forward and Backward Algorithms:** These algorithms are used to calculate the probability of a given sequence being generated by an HMM. Most of the problems solved by these algorithms come under the EVALUATION part of HMMs.

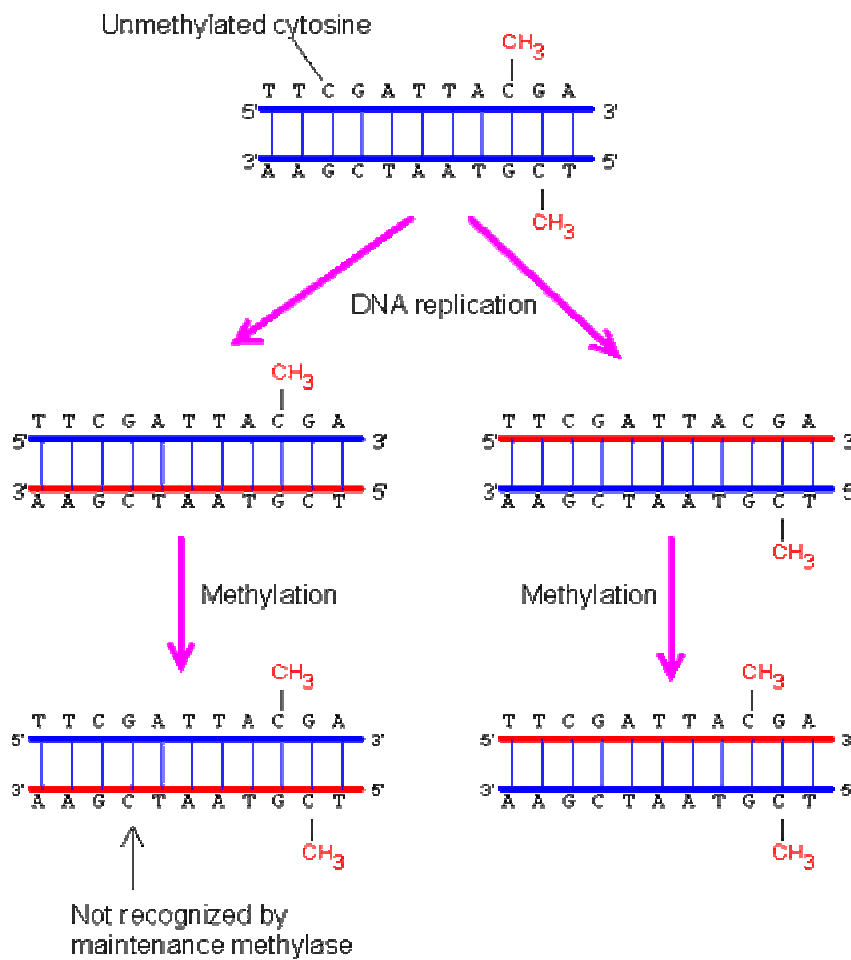
e.g.: In the casino player, finding the probability of a sequence of rolls or finding whether a particular roll was generated by a fair or loaded die.

**2. A Biological example where HMM is applicable (CpG Islands)**

All of our examples till now involved the Casino player so lets see how to make use of HMMs in DNA sequences.

The kind of information we will be considering is known as “epigenetic information”. We first discuss a process known as methylation.

**DNA methylation:** This is a modification to the DNA molecules that adds a methyl (-CH<sub>3</sub>) group to a particular nucleotide. These kind of modifications also get inherited when cell division takes place.



**Fig 1: Methylation process**

**Notes about methylation:**

- In mammals, methylation occurs in CG nucleotides. (CG has a property that its reverse complement is also CG so while copying a DNA the same pattern of methylation can be maintained)
- Methylation enzymes are used in the process of methylation.
- Most of the time, important bookkeeping genes are not methylated because methylated genes get silenced after the process.

**CpG Islands:**

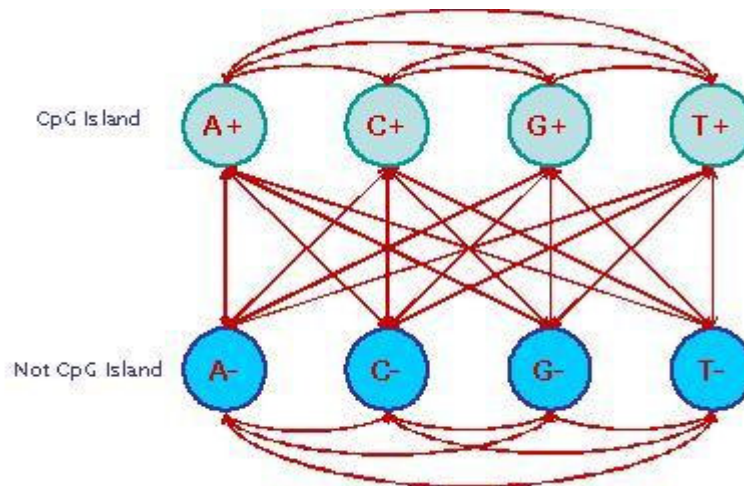
CG nucleotides (denoted as CpG to distinguish them from CG base pair ) often mutate to TG when methylated. So the CG pairs become rare in areas of high methylation. So, there is a high concentration of CG in important genes since the CG here is not methylated. These regions are known as CpG islands. So, one way to identify important genes is the find regions in the genome of high CpG content.

So the problem we are looking to solve is : **How to detect the CpG Islands computationally?**

Let us try out different models for this problem and see which of them models it accurately enough to get the best results.

**Model 1:**

We have 4 states corresponding to the 4 nucleotides for the CpG island region and four more states for the non- CpG island region as shown in the figure. So the CpG states are represented by a (+) while the the non CpG states are represented by a (-).



**Fig 2 : Model for CpG Island**

We consider a known CpG island from the genome and count how many times A,C,G,T come next to A and find the transition probabilities from A+. We do this computation for the other states and get all the transition probabilities for the HMM.

For example, here is a table of transition probabilities generated for the CpG island states.

+	A	C	G	T
A	.180	.274	.426	.120
C	.171	.368	.274	.188
G	.161	.339	.375	.125
T	.079	.355	.384	.182

**Fig 3 : Table of transition probabilities**

So we have in all 32 (16 \* 2) transition states without considering the transitions for the CpG region to non CpG region.

Now given this model, a simple way to find out if a given region is part of CpG island or not, we find the log likelihood by calculation the ratio of the log odds of the two probabilities for the CpG and non CpG regions.

$$L(u,v) = \log (P(uv|+) / P(uv|-) )$$

So, now if this value > 0, then we can say that the region is a CpG island else if value is less than 0, we can consider it to be a non CpG island. This is not a very accurate model since it does not consider the transitions between the CpG and non CpG states.

**Model 2:**

Let

P<sub>++</sub> represent the probability of staying within the CpG island states

P<sub>--</sub> represent the probability of staying within the non CpG island states.

Now each of the transition probabilities can be modified by these probabilities. For example, each transition probability among states in the CpG island will be scaled by P<sub>++</sub>.

i.e  $a_{(A+G+)} = P_{++} * a_{(A+G+)}$

Now let us consider transitions among the CpG and non CpG states.

So the idea is when moving from a state in a CpG island to one in non CpG there are two factors to account for:

1. The probability of going to the non CpG region (1- P<sub>++</sub>)

2. The probability of occurrence of that state in the non CpG island region.

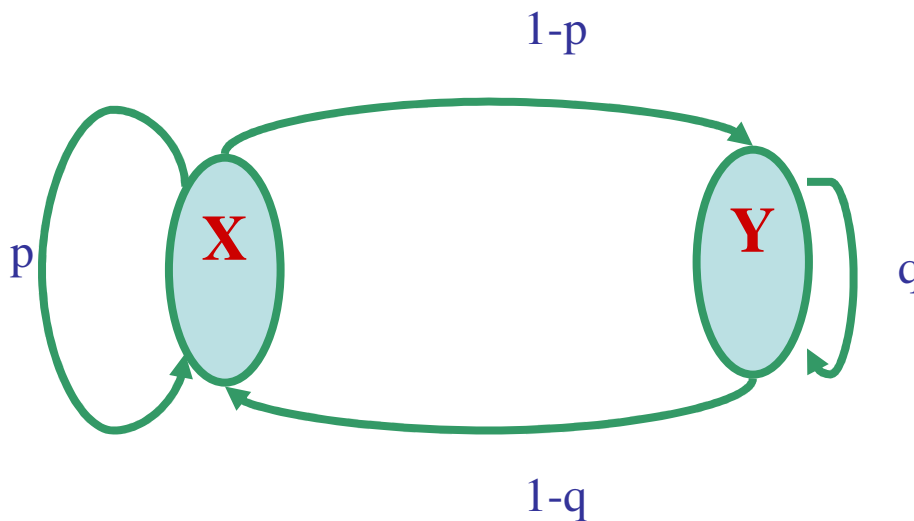
For example,

$$a_{(A+A-)} = (1 - P_{++}) * q_{A-} \text{ (this is the proportion of A in the non CpG region)}$$

Similarly we can calculate the transition probabilities to the non CpG regions. But we need some way of estimating the value of  $P_{++}$  and  $P_{--}$  to find out these transition probabilities. To find this out, lets just take a quick digression to the length of distribution of each state in an HMM.

**Geometric Distribution of State Duration**

Given an HMM as shown below,



**Fig 4: HMM example to describe the geometric distribution of states**

$P[I_X = 1] = (1-p)$  (Probability of remaining in state X for a single time interval)  
 And  $P[I_X = 2] = p(1-p)$ .

So, we can calculate the average length of this state as  $(1/1-p)$  which shows that the length distribution of each state tends to fall geometrically.

We can now estimate the average length of a CpG island and the average length between two CpG islands to compute the required probabilities.

Given such a model we can use the HMM algorithms to compute the following:

1. Viterbi : To compute the best parse for a given sequence which can give us whether a particular nucleotide is in a CpG region or not given the complete DNA sequence.
2. Posterior Decoding: If we want to find a more locally optimum prediction for a state given the sequence using forward/backward algorithms.

### **3. The “LEARNING” problem in HMM**

#### **Motivation**

Suppose we come up with a newly sequenced genome of an organism for which we do not have much prior information. So, for example we do not know the CpG island regions in the genome but would surely like to train the parameters of our model to learn these values.

So the learning problem in HMM involves updating the parameters  $\theta$  of the model to maximize the quantity  $P(x|\theta)$ .

#### **Learning Scenarios**

There are two different type of learning scenarios depending on the type of data available with us. These are in some ways related to supervised and unsupervised learning scenarios in machine learning.

##### **a. When the right answer is known!**

These are scenarios when we are given a set of training sequences along with labeled parses for these sequences. So the data set consists of a sequence  $X = x_1 x_2 \dots x_n$  for

which the true parse  $\pi = \pi_1 \pi_2 \dots \pi_n$  is known.

In this case, the new parameters  $\theta$  maximize the quantity  $P(x, \pi | \theta)$ .

So, given labeled data, we can find out a very intuitive way to define the transition and emission probabilities of our HMM.

If  $A_{kl}$  represents the number of times  $k \rightarrow l$  transitions occur in  $\pi$  and

$E_k(b)$  represents the number of times state  $k$  in  $\pi$  emits  $b$  in  $x$ ,

The maximum likelihood parameters can just be the frequency count from this data.

So,  $a_{kl} = A_{kl} / \sum_i A_{ki}$  and  $e_k(b) = E_k(b) / \sum_c E_k(c)$

The drawback of this scheme is that the parameters might be over fitted for the given training data so when new and more generalized data is provided, the model will not work well in those cases.

Also, some of the probabilities might end up with a value of zero which is incorrect and must be set to a very small value in these cases.

So, a way around this is to use “pseudo counts” and add them to the values in the equations above which might depend on the prior beliefs we have regarding these values.

e.g:  $A_{kl} = (\text{the number of times } k \rightarrow l \text{ transitions occur in } \pi) + n_{kl}$

### **b. When the right answer is not known!**

These represent scenarios when we are given a set of training examples but without labels. We thus have no knowledge of the true parse of the sequences and have to learn the parameters just by using all the sequences provided.

So the  $A_{kl}$  and  $E_k(b)$  values cannot be directly computed as was possible in the previous case.

The EM or Expectation Maximization algorithm is a general method of finding the maximum-likelihood estimate of the parameters of an underlying distribution from a given data set when the data is generally incomplete or missing. For our scenario the EM algorithm can be briefly outlined as:

1. Estimate  $A_{kl}$ ,  $E_k(b)$  in the training data
2. Update  $\theta$  according to  $A_{kl}$ ,  $E_k(b)$
3. Repeat steps (1) and (2) until convergence

Let us see two solutions to the problem.

### **Baum Welch Algorithm:**

This is a variant of the EM algorithm and its details are given below.

#### **Algorithm steps**

- i. Initialize the parameter models arbitrarily
- ii. Given a  $\theta_{curr}$  to work with, we have to estimate the values for  $A_{kl}$  and  $E_k(b)$  as in the previous case. The idea is to find the probability of a  $k \rightarrow l$  state transition at a particular position  $i$  in the parse corresponding to the sequence.

At each position  $i$  of sequence  $x$ , find probability transition  $k \rightarrow l$  is used:

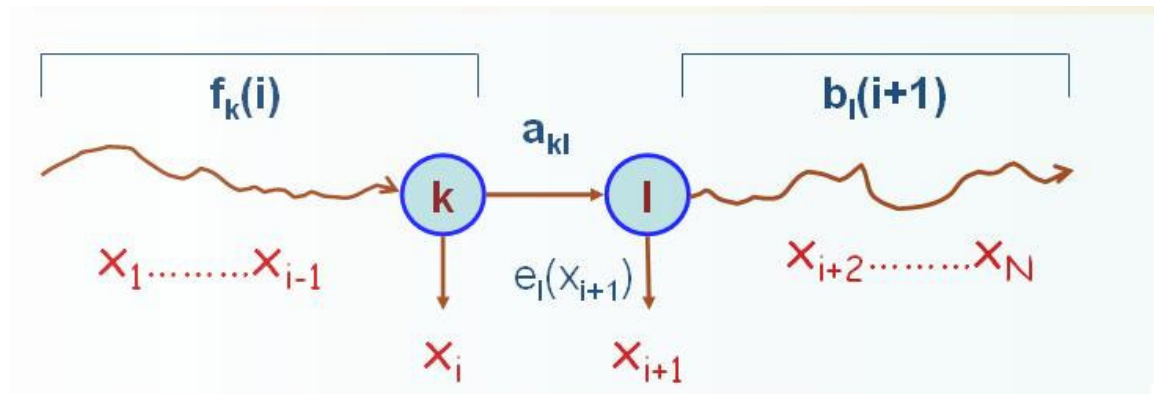
So,

$$P(\pi_i = k, \pi_{i+1} = l | x) =$$

$$[1/P(x)] \times P(\pi_i = k, \pi_{i+1} = l, x_1 \dots x_N) = Q/P(x) \quad (\text{By Conditional Probability rule})$$

By using the Markov property and separating out the terms, we get terms representing forward and backward computations. The figure explains the formula well describing the transition from k->l being composed of all the previous states and emissions followed by the transition from k to l with the emission at l and then all remaining state transitions and emissions.

$$\text{So: } P(\pi_i = k, \pi_{i+1} = l | \mathbf{x}, \theta) = \frac{f_k(i) a_{kl} e_l(x_{i+1}) b_l(i+1)}{P(\mathbf{x} | \theta_{curr})}$$



**Fig 5: Computations involved in Baum Welch Algorithm**

Similarly ,

$$E_k(\mathbf{b}) = [1/P(\mathbf{x}|\theta)] \sum_{\{i|x_i=b\}} f_k(i) b_k(i)$$

So, compute forward, backward parts and using above formulae, compute A and E

iii. The next step is to update the parameter model using these values and computing the log likelihood i.e  $P(\mathbf{x} | \theta_{new})$

iv. Repeat steps ii. and iii. till the difference in the results does not vary by much.

**Comments on the algorithm**

1. The algorithm requires the same running time as the forward and backward algorithms take for computation i.e is  $O(k^2 N)$
2. Since this is based on the EM algorithm, it is guaranteed to improve the log likelihood result during each iteration.
3. The algorithm suffers from the drawback that it might end up at a local maximum instead of finding the optimum global maximum result.

### **Viterbi Algorithm**

Another simple approach to solve the problem is using the Viterbi algorithm. Our problem was mainly estimating the values of  $A_{kl}$ ,  $E_k(b)$  without knowing the actual parse for the given sequences. The Viterbi actually does give us the most likely parse for a sequence.

So the steps for the algorithm are as follows.

- i. Find the best parse  $\pi^*$  using Viterbi.
- ii. Now, similar to the first case, estimate the values for  $A_{kl}$ ,  $E_k(b)$  using the frequency count.
- iii. Update the values of the HMM model based on these values

Repeat the steps ii and iii till convergence.

### **Comments on the algorithm**

This algorithm gives a more poor result as compared to Baum Welch because the quantity that it is maximizing is not  $P(x | \theta_{new})$  but  $P(x | \theta_{new}, \pi^*)$ . But it can be used as an alternative since Viterbi is very easy to implement and provide immediate results.

## **4. Variants of HMM**

Let us see some drawbacks/weakness of HMM and how we could improve them by different variations of the model.

### **A. Memory less HMM**

One of the most basic assumptions made by HMM is that the current state parameters depend only on the previous state of the HMM and independent of what happened before that. Thus an HMM tends to forget all previous states that have been traversed. Depending on what is being modeled in real world this may not always be the case.

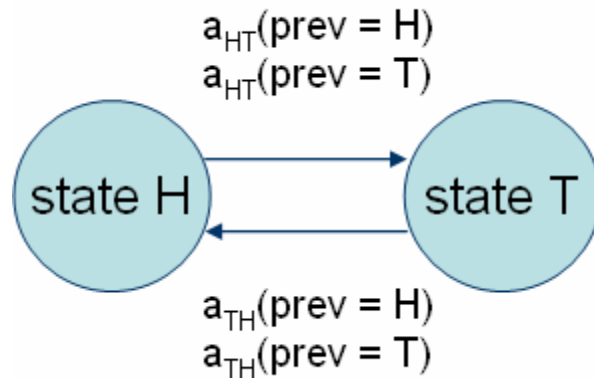
### **Examples**

When using HMM in Natural Language processing(word representing a state say), having a larger context (keeping track of 2-3 words on either side of the given word) provides richer results as compared to knowing only the previous word.

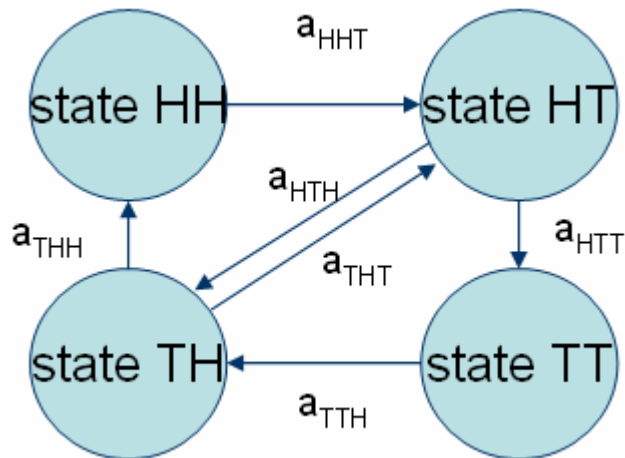
A biological example could be RNA secondary structures which have longer range interactions with symbols which cannot be suitably modeled using HMMs.

So higher order markov models are required. A second order HMM can be described as a first order HMM with  $k^2$  states. Possible representations for second order HMMs are shown below. The next state depends on two previous input states.

In the first representation, moving from the state H to t using the  $a_{HT}$  (prev = H) transition is equivalent to a  $a_{HHT}$  transition from HH to HT state in the second representation. Similarly we could find equivalence in both the representations.



**Fig 6: Second order HMM (1)**



**Fig 7 : Second order HMM(2)**

**B. Geometric Distribution of states**

This problem of HMM was making it difficult to model the distribution of exons and introns in the genes. So different solutions to this problem like chaining of states and obtaining a negative binomial distribution have been tried out.

The details of the problem and its solutions will be covered in the next lecture. Also, a much more powerful graphical model CRF (Conditional random fields) discovered very recently (around 1998) is emerging to be quite useful in the field of Bioinformatics and will be the main topic of discussion next time.