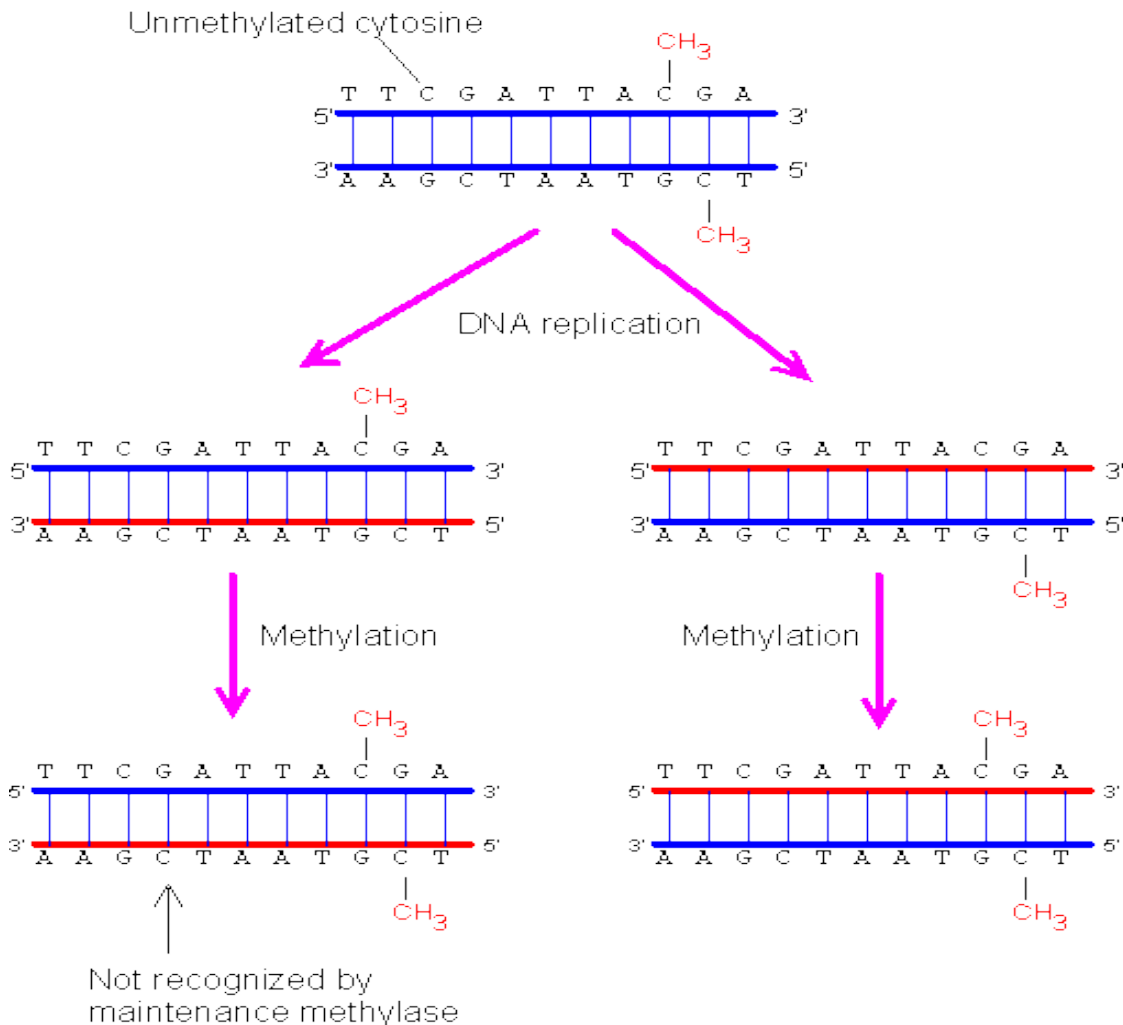


CpG Islands in DNA Sequences

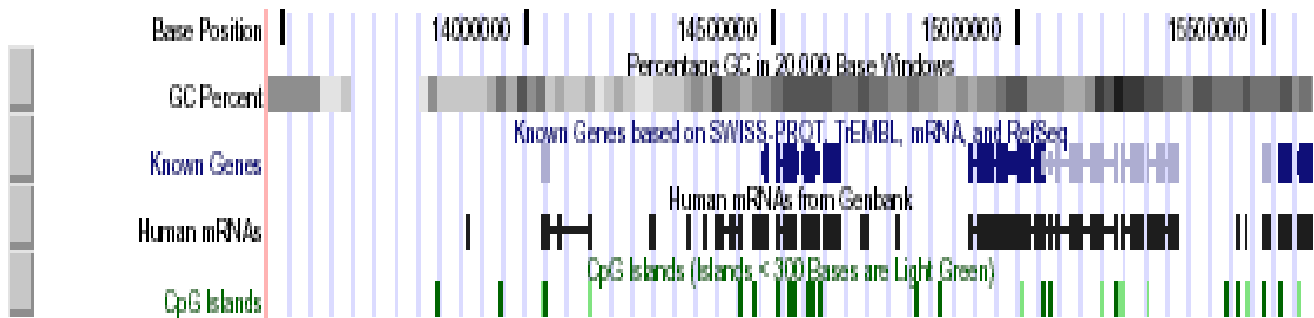
Detecting CpG islands are an example use case of Hidden Markov Models (HMM). In a genome, a CG dinucleotide often mutates to a TG dinucleotide due to methylation. There is therefore a decrease in the probability of seeing a CG pair (and an increased probability of seeing other pairs) in these methylated regions which are called CpG islands.

There are several biological reasons to identify CpG islands. Methylation is often suppressed around the promoters and start regions of a gene. Additionally, Methylation silences genes in the same region.

In the picture below the C bases with CH₃ bonded are methylated and will with higher probability mutate into Ts in the future.

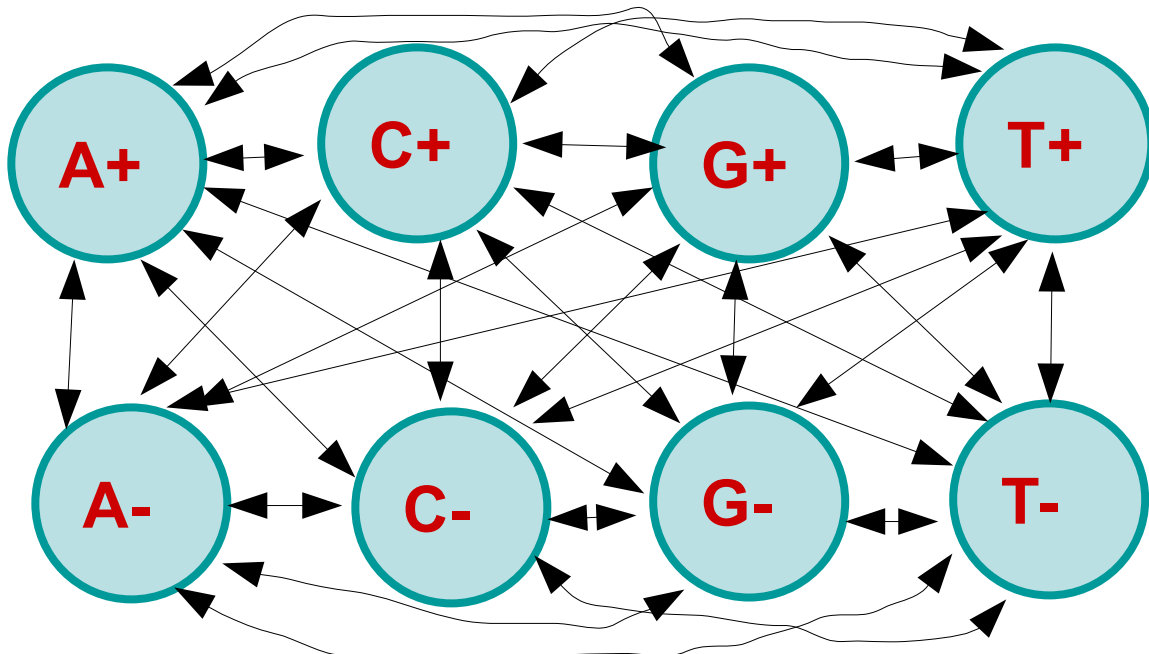


In the chart below the CpG islands are clustered around the start of genes:



Hidden Markov Models can be used to detect these CpG islands computationally. In the last lecture we considered the example of a sequence of dice roles which were generated by two dice (although you don't know a priori which dice was used for which role). We then used a Hidden Markov model to predict which dice were used for which regions of dice roles.

Similarly, given a DNA sequence we can use a Hidden Markov model to predict which regions are methylated (CpG Islands) and which are not since CpG islands have different concentrations of nucleotides. Initially it might seem like we should model the DNA sequence with two states, the first state corresponding to not being in a CpG island and the second the state the opposite. However, for a DNA sequence, the emission probability of nucleotide is dependent on the previous emission, not the previous state (CpG island/not a CpG island) and hence we need more than two states. We need four states (one for each nucleotide) to model being in a CpG Island and four states to model not being a CpG region. The HMM then looks as follow:



Note that in this model the emission probabilities in each state are binary. For example, state A+ only emits an A and T- only emits a T. Below is an example table of emission probabilities for the + (CpG island) and - (non-CpG island) states. Note that in this example there is zero probability of switching between a CpG island and a non-CpG island.

+	A	C	G	T	-	A	C	G	T
A	.180	.274	.426	.120	A	.300	.205	.285	.210
C	.171	.368	.274	.188	C	.233	.298	.078	.302
G	.161	.339	.375	.125	G	.248	.246	.298	.208
T	.079	.355	.384	.182	T	.177	.239	.292	.292

How do you tell a CpG Island from a non-CpG Island?

If we have the transition probabilities, the simplest way to determine whether two nucleotides in our sequence, uv , are in a CpG Island is just to compare the transition probabilities of u to v between the CpG Island states and the non-CpG Island states. That is, if $P(uv | +) / P(uv | -) > 1$. If we take the log of both sides then we get:

$$\log[P(uv | +) / P(uv | -)] > 0$$

This can be logically extended to the whole sequence by summing the ratios of every transition in the sequence to get:

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

$$\sum_j L(x_j, x_{j+1}) > 0$$

Where if the summation is greater than zero the sequence is probably a CpG Island.

Transitions Between + and – States

The transition probabilities between + and – states effect the length of detected CpG Islands in the sequence. If the probability of staying in a + state is p than the probability of transitioning from the + state to the – state is $(1-p)$. Hence, if we start in a + state at the first nucleotide in a sequence of length x , the probability of transitioning to a – state is $(1-p)$. The probability of staying in the + state the first transition and transitioning to a – state in the second transition is than $p(1-p)$. Following this pattern, the probability of staying in + states for x transitions is $p^{x-1} (1-p)$. This is a geometric distribution so the expected length of the + region is $1/(1-p)$.

Analyzing a Sequence

Given a DNA sequence, the Viterbi algorithm described in the last class can be used to find the most probable CpG Islands. Another technique is to use the forward and backward algorithms presented last class to find :

$$P(x_i \text{ is in CpG island}) = P(\pi_i = A^+ | x)$$

These probabilities can then be used in Posterior decoding to find the most probable state for each emission, independent of the rest of the sequence. This is defined as:

$$\hat{\pi}_i = \operatorname{argmax}_k P(\pi_i = k | x)$$

Where π_i is the state and x is the sequence. Posterior decoding has the down side of creating CpG islands which are often too short and choppy since for each nucleotide we are picking the state which has maximum probability independent of the state chosen for the nucleotide's neighbors.

How do we Pick the Transition Probabilities

Given a sequence of DNA we may not know the correct transition probabilities between states in the HMM for this type of genome. There are two options. One thing we can do is find a genome that is biologically similar to the genome we are studying (e.g. a humans DNA is similar to that of a Chimpanzees) and use its transition probabilities as a substitute. This sometimes works in practice.

Another possibility is to learn the transition probabilities. That is, we can estimate the parameters of our Hidden Markov model using training data.

Learning Parameters

There are two learning scenarios we will look at:

1. Supervised learning – We have sample DNA sequences which are annotated with which portions are CpG Islands.
2. Unsupervised learning – We have sample DNA sequences but we do NOT know which regions are CpG Islands

Our goal in both of these cases is to find parameters θ such that $P(x|\theta)$ is maximized.

Supervised Learning

Given a DNA sequence and the states π from which each nucleotide was emitted, we can define the transition probability from state l to state k as the number of times our training sequence went from state l to k over the total number of times our training sequence transitioned out of state l . Similarly, we can define the emission probability of emitting nucleotide x in state s as ratio of the number of times s emitted x over the total number of times s emitted any nucleotide. These values are then the maximum likelihood parameters for the transition and emission probabilities.

$A_{kl} = \# \text{ times } k \rightarrow l \text{ transition occurs in } \pi$

$E_k(b) = \# \text{ times state } k \text{ in } \pi \text{ emits } b \text{ in } x$

$$a_{kl} = \frac{A_{kl}}{\sum_i A_{ki}} \quad e_k(b) = \frac{E_k(b)}{\sum_c E_k(c)}$$

There are a few pitfalls which need to be avoided when using this approach. If there is little data available, or if the data is not representative of the rest of the genome, then the parameters learned will either be overfitted or skewed. The second pitfall is that zero probabilities are not handled gracefully. If some transition or emission never occurs in the sample data then it will be given zero probability in this scheme. Using these parameters, any parse of new DNA sequence which does contain a transition or emission

not seen before will receive a probability of zero, regardless of how promising the parse was modulo that one errant transition/emission.

A solution to this problem is adding initial pseudo counts to each transition and emission count so that there is never a zero probability transition or emission. Below r is the pseudo count being added.

$$A_{kl} = \# \text{ times } k \rightarrow l \text{ transition occurs in } \pi + r_{kl}$$

$$E_k(b) = \# \text{ times state } k \text{ in } \pi \text{ emits } b \text{ in } x + r_k(b)$$

Additionally, pseudo counts can be used to represent prior beliefs about the transition and emission probabilities. For example, if we have a very strong belief about what the value of what A_{kl} is, than making r_{kl} larger decreases the influence of our training data on the transition probability.

Unsupervised Learning

Given our training sequence x , we do not know which portions are CpG islands and which portions are not. Our approach to find the transition probabilities is therefore a two step process. First we estimate what A_{lk} and $E_k(b)$ are. Then we update the parameters of our model and repeat this process until our parameters converge.

Holding our parameters fixed, we can use the forward and backward algorithms presented last lecture to calculate the probability of switching from state l to state k at position i of the sequence.

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

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

where:

$$\begin{aligned}
 Q &= P(x_1 \dots x_i, \pi_i = k, \pi_{i+1} = l, x_{i+1} \dots x_N) = \\
 &= P(\pi_{i+1} = l, x_{i+1} \dots x_N \mid \pi_i = k) P(x_1 \dots x_i, \pi_i = k) = \\
 &= P(\pi_{i+1} = l, x_{i+1} x_{i+2} \dots x_N \mid \pi_i = k) f_k(i) = \\
 &= P(x_{i+2} \dots x_N \mid \pi_{i+1} = l) P(x_{i+1} \mid \pi_{i+1} = l) P(\pi_{i+1} = l \mid \pi_i = k) f_k(i) = \\
 &= b_l(i+1) e_l(x_{i+1}) a_{kl} f_k(i)
 \end{aligned}$$

$$f_k(i) a_{kl} e_l(x_{i+1}) b_l(i+1)$$

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

A_{lk} can then be defined as the expected number of these k to l transitions in the sequence given the current parameters. This is just the summation of the probability of the transition occurring at each index i in the sequence.

$$\begin{aligned}
 A_{kl} &= \sum_i P(\pi_i = k, \pi_{i+1} = l \mid x, \theta) \\
 &= \sum_i \frac{f_k(i) a_{kl} e_l(x_{i+1}) b_l(i+1)}{P(x \mid \theta)}
 \end{aligned}$$

Through the exact same reasoning the equation for the emission probabilities is found.

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

Baum-Welsh Algorithm

An unsupervised learning algorithm using the iterative 1)expectation 2)maximization approach.

(The algorithm as given in Durbin)

Initialization: Pick arbitrary model parameters.

Recurrence:

Set all the A and E variables to their pseudocount values r

For each sequence $j=1\dots n$

Calculate $f_k(i)$ for sequence j using the forward algorithm

Calculate $b_k(i)$ for sequence j using the backward algorithm

Add the contributions of sequence j to A and E

Calculate the new model parameters using the same equations used in the supervised learning section.

Calculate the log likelihood of the model.

Termination: Stop if the change in log likelihood is less than some predefined threshold of the maximum number of iterations is exceeded.

Baum-Welsh finds locally optimal parameters (from where they were initialized) not the globally optimal.