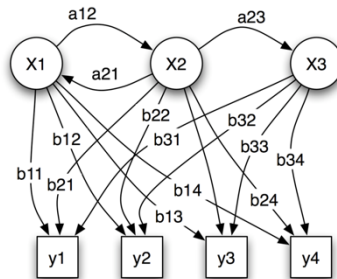


# Hidden Markov Models



Scott Pegg, Ph.D.

BMI203 February 7, 2018

© 2006 Scott Pegg

## Where we're going today

- Markov & his chains
- What's a Hidden Markov Model?
- What does it have to do with biology?
- What can I do with one? How?
- I'm confused, where do I learn more?

© 2006 Scott Pegg

## Markov who?



Andrei Andreyevich Markov  
1856 - 1922  
St. Petersburg, Russia



Grigory Yefimovich Rasputin  
1871 - 1916  
St. Petersburg, Russia

© 2006 Scott Pegg

## Markov Process

A process in which the state at time  $t$  depends upon the state at times  $t-1, t-2, \dots, t-k$

$k^{\text{th}}$  order Markov process

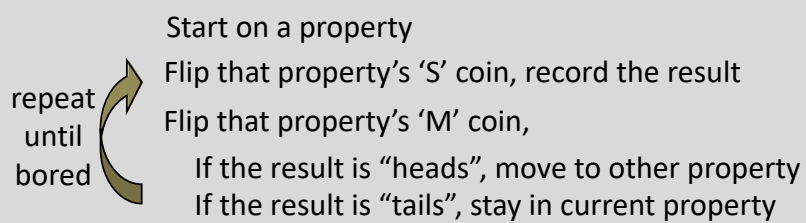
Most often, we're interested in a first order model

aka 'Markov chain'

A set of states  $S = \{S^0, S^1, \dots, S^N\}$   
such that  
$$P(S^{t+1} | S^t) = P(S^{t+1} | S^0 S^1 \dots S^t)$$

© 2006 Scott Pegg

# A first-order Markov process

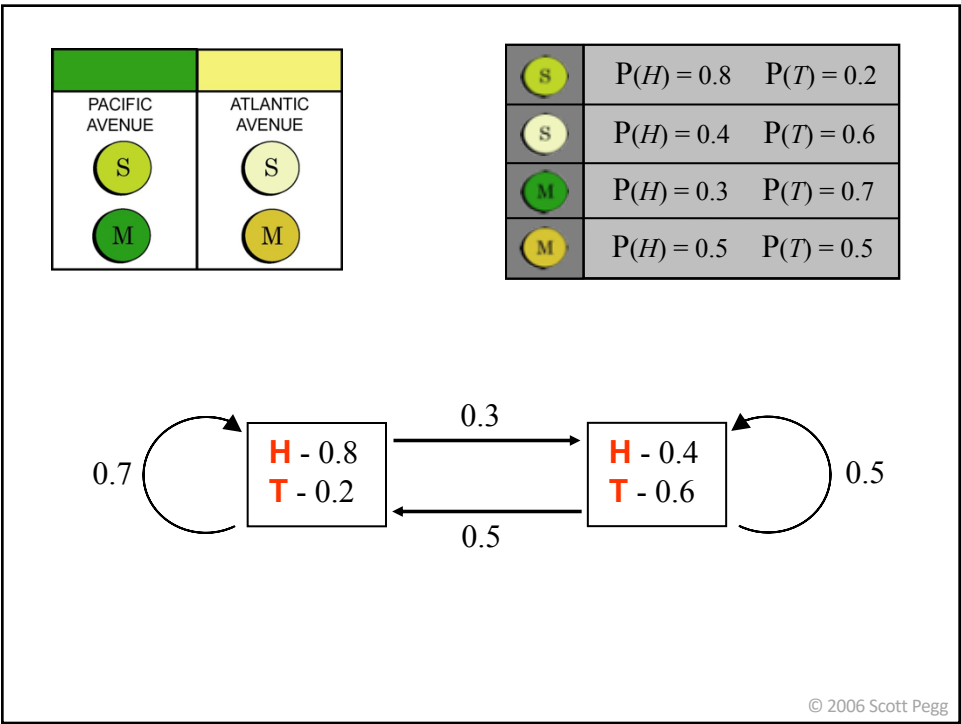


	PACIFIC AVENUE	ATLANTIC AVENUE
	S	S
	M	M

S	$P(H) = 0.8$ $P(T) = 0.2$
S	$P(H) = 0.4$ $P(T) = 0.6$
M	$P(H) = 0.3$ $P(T) = 0.7$
M	$P(H) = 0.5$ $P(T) = 0.5$

© 2006 Scott Pegg

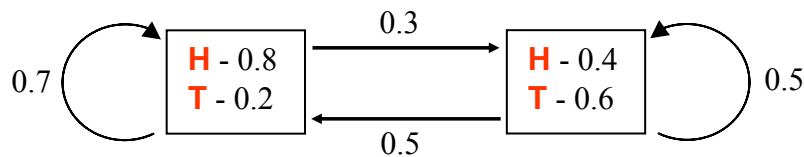


© 2006 Scott Pegg

At the end of the game, I have a string of symbol observations

...THHTTHHTHTHTHTHTHTHT...

But I no longer remember what state (Monopoly property) each symbol was emitted from...



The states are what's "hidden" in a hidden Markov Model

© 2006 Scott Pegg

## HMMs in Biology

In biology, we often see strings of symbols

GCACCGTTAGGACAGGA  
YLRNGYITSGYPLMFLHLL

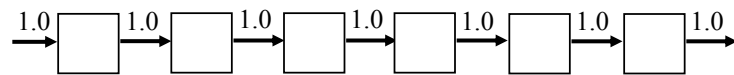
And often in related groups

GCACCGTTAGGACAGGA  
GGACCATTACGGCGGCA  
CCAGCGTATCCGCAACA

We build HMMs which could generate them...

© 2006 Scott Pegg

<b>G</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>G</b>	<b>T</b>	<b>C</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>G</b>	<b>C</b>	<b>C</b>	<b>T</b>	<b>T</b>	<b>G</b>
<b>G</b>	<b>T</b>	<b>A</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>C</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>T</b>	<b>G</b>
<b>C</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>A</b>	<b>G</b>



The easiest architecture is to have each column be a state.

© 2006 Scott Pegg

<b>G</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>G</b>	<b>T</b>	<b>C</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>G</b>	<b>C</b>	<b>C</b>	<b>T</b>	<b>T</b>	<b>G</b>
<b>G</b>	<b>T</b>	<b>A</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>C</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>T</b>	<b>G</b>
<b>C</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>A</b>	<b>G</b>

<b>C(A)</b>	0	5	1	5	4	0
<b>C(T)</b>	0	0	0	1	2	0
<b>C(C)</b>	2	1	2	0	0	0
<b>C(G)</b>	3	0	3	0	0	6

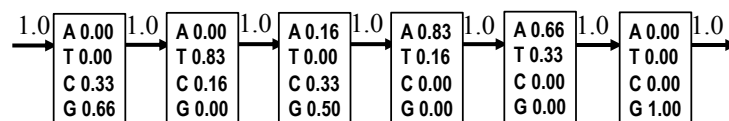
© 2006 Scott Pegg

<b>G</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>G</b>	<b>T</b>	<b>C</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>G</b>	<b>C</b>	<b>C</b>	<b>T</b>	<b>T</b>	<b>G</b>
<b>G</b>	<b>T</b>	<b>A</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>C</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>T</b>	<b>G</b>
<b>C</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>A</b>	<b>G</b>

<b>P(A)</b>	0.00	0.00	0.16	0.83	0.66	0.00
<b>P(T)</b>	0.00	0.83	0.00	0.16	0.33	0.00
<b>P(C)</b>	0.33	0.16	0.33	0.00	0.00	0.00
<b>P(G)</b>	0.66	0.00	0.50	0.00	0.00	1.00

© 2006 Scott Pegg

<b>G</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>G</b>	<b>T</b>	<b>C</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>G</b>	<b>C</b>	<b>C</b>	<b>T</b>	<b>T</b>	<b>G</b>
<b>G</b>	<b>T</b>	<b>A</b>	<b>A</b>	<b>A</b>	<b>G</b>
<b>C</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>T</b>	<b>G</b>
<b>C</b>	<b>T</b>	<b>G</b>	<b>A</b>	<b>A</b>	<b>G</b>



© 2006 Scott Pegg

G	T	G	A	A	G
G	T	C	A	A	G
G	C	C	T	T	G
G	T	-	A	A	G
C	T	-	A	T	G
C	T	-	A	A	G

What if there are gaps?

Does a gap represent an insertion or a deletion?

© 2006 Scott Pegg

G	T	G	A	A	G
G	T	C	A	A	G
G	C	C	T	T	G
G	T	-	A	A	G
C	T	-	A	T	G
C	T	-	A	A	G

Instead of one state  
per column, we now  
have three!

-- 1.00

delete state

A 0.00  
T 0.00  
C 0.66  
G 0.33

match state

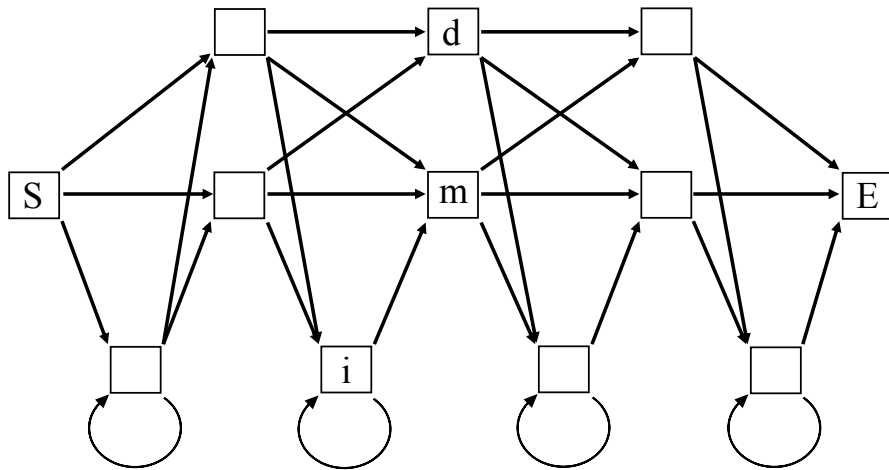
A 0.25  
T 0.25  
C 0.25  
G 0.25

insert state

© 2006 Scott Pegg



## A typical bioinformatics HMM



## Three Fundamental Questions

Given a sequence of symbols and a HMM,

1. What's the most probable sequence of transitions and emissions?

'decoding'

2. How likely is this sequence given the HMM?

'likelihood'

3. How should the transition and emission probabilities be updated?

'learning'

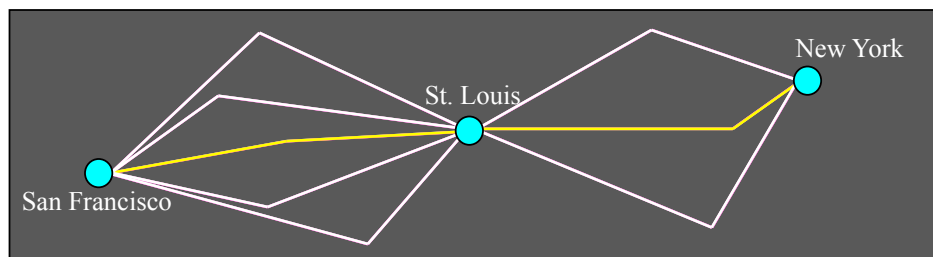
© 2006 Scott Pegg

## The 'decoding' question

What's the most probable sequence of transitions and emissions to produce the observed sequence of symbols?

### The Viterbi algorithm

For any state at time  $t$ , there is only one most likely path to that state.



When calculating the transitions from this state to states at time  $t+1$ , one can discard the less likely paths.

© 2006 Scott Pegg

## The Viterbi Algorithm

Let  $\mathbf{X}$  be the sequence of symbols  $x_1 x_2 \dots x_L$

Let  $v_k(i)$  = probability of the most probable path for sequence  $x_1 x_2 \dots x_i$  that ends in state  $k$

		symbol index ( $i$ )				
		0				L
state ( $k$ )	0	1.0	$v_0(1)$			
		0.0				
		0.0				
		0.0				
		0.0				
S		0.0				$v_S(L)$

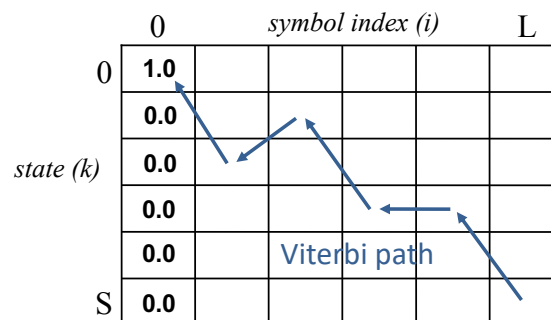
© 2006 Scott Pegg

$$v_j(i+1) = e_j(x_{i+1}) \max_{k \in S} \{v_k(i) a_{kj}\}$$

where

$e_j(x_i)$  = probability of emitting symbol  $x_i$   
from state  $j$

$a_{kj}$  = probability of a transition from state  $k$   
to state  $j$



© 2006 Scott Pegg

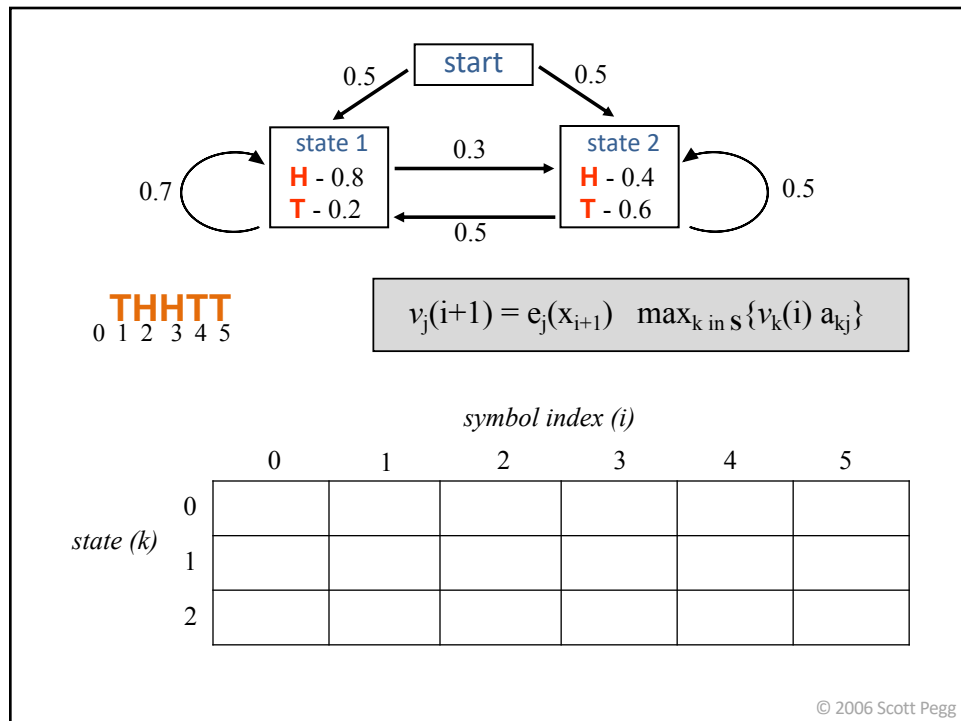
## The Viterbi Algorithm

At the end, one has the final probability of the sequence given the most probable path,

$$P(\mathbf{X}|\pi^*) = \max_{k \in S} \{v_k(L) a_{k\text{end}}\}$$

The most probable state path  $\pi^*$  is recovered by simply tracing back along saved state pointers.

© 2006 Scott Pegg



		symbol index ( $i$ )					
		0	1	2	3	4	5
state ( $k$ )	0	1.0	0.0	0.0	0.0	0.0	0.0
	1	0.0	0.10	0.12	0.0672	0.009408	0.001317
	2	0.0	0.30	0.06	0.0144	0.012096	0.003629

underflow errors!

The solution is to convert probabilities to logarithms.

**Step 1:**  $v_{\text{start}}(0) = 0$  for all other states  $k$ ,  $v_k(0) = -\infty$

**Step 2:**  $v_j(i+1) = \text{Log } e_j(x_{i+1}) + \max_{k \in S} \{v_k(i) + \text{Log } a_{kj}\}$

**Step 3:**  $S(\mathbf{X}|\pi^*) = \max_{k \in S} \{v_k(L) + \text{Log } a_{\text{kend}}\}$

© 2006 Scott Pegg

## The 'decoding' answer

So what does answering the 'decoding' question give us?

It assigns each observed symbol to a state

Aligning the states aligns the symbols

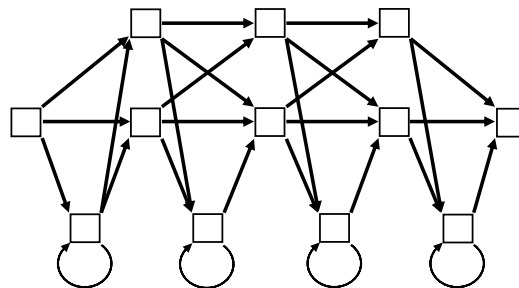
<b>T L F A - G P G</b>
<b>E L F A G G P C</b>

States can provide gap information

© 2006 Scott Pegg

## The 'likelihood' Question

Given a sequence of symbols and a HMM, how likely is this sequence given the HMM?



Find all of the combinations of states (ie. paths) and emissions that could generate our sequence of symbols and calculate the sum of their probabilities.

But there are an exponential number of paths!

© 2006 Scott Pegg

## The 'forward' Algorithm

Given the sequence of symbols  $\mathbf{X} = \{x_1, x_2, \dots, x_L\}$ ,  
let  $f_k(i)$  = the probability of having emitted the prefix  
 $\{x_1, x_2, \dots, x_i\}$  and reaching state  $k$ .

		<i>symbol index (i)</i>				
		0				L
<i>state (k)</i>	0	<b>1.0</b>	$f_0(1)$			
		<b>0.0</b>				
		<b>0.0</b>				
		<b>0.0</b>				
		<b>0.0</b>				
S		<b>0.0</b>				$f_S(L)$

© 2006 Scott Pegg

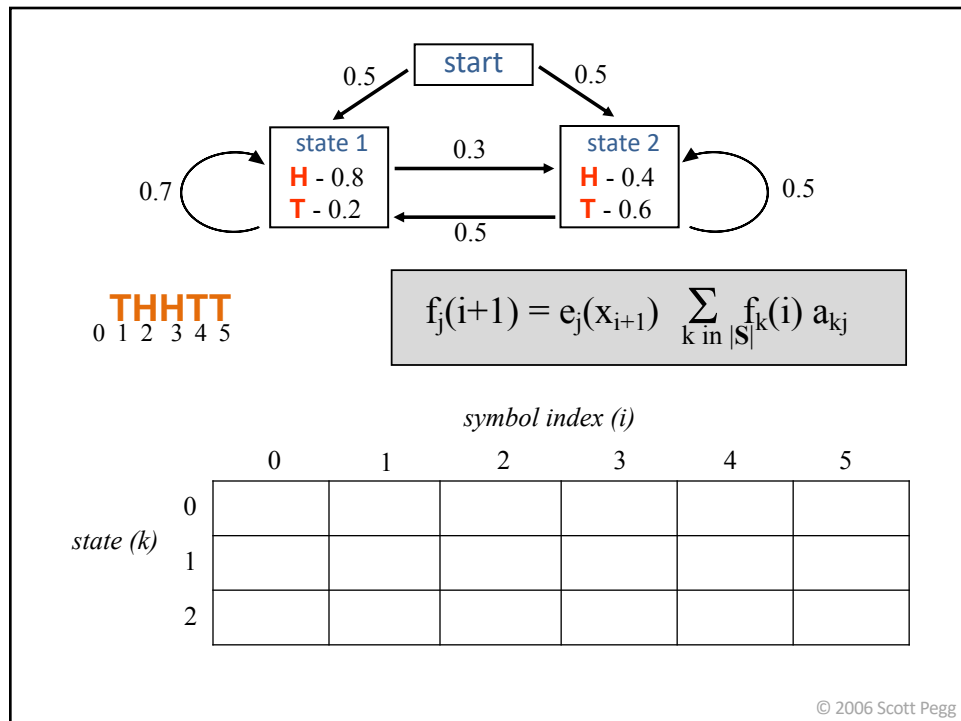
## The 'forward' Algorithm

At the end of the calculation,

$$P(\mathbf{X}) = \sum_{k \text{ in } |S|} f_k(L) a_{\text{kend}}$$

= probability of the sequence being  
produced by the model

© 2006 Scott Pegg



## The 'backward' Algorithm

Given the sequence of symbols  $\mathbf{X} = \{x_1, x_2, \dots, x_L\}$ ,  
 let  $b_k(i)$  = the probability of having emitted the suffix  
 $\{x_{i+1}, x_{i+2}, \dots, x_L\}$  and reaching state  $k$ .

	symbol index (i)				
	0				L
state (k)	0	$b_0(0)$			$a_{0S}$
					$a_{1S}$
					$a_{2S}$
					.
					.
S				$b_S(L-1)$	$a_{SS}$

© 2006 Scott Pegg

## The 'backward' Algorithm

Fill in the table backwards, using the recurrence relation

$$b_j(i) = \sum_{k \text{ in } |S|} a_{jk} e_k(x_{i+1}) b_k(i+1)$$

At the end

$$P(\mathbf{X}) = \sum_{k \text{ in } |S|} a_{\text{start } k} e_k(x_1) b_k(1)$$

= probability of the sequence being produced by the model

© 2006 Scott Pegg

## Which way do I go?

For  $P(\mathbf{X})$ , you can go forward or backward.

Sometimes, however, you want to know which state was the most likely to have produced any given symbol. To figure this out, we go both ways.

We want to know

$$P(\pi_i=k \mid \mathbf{X}) = \text{the probability of state } i \text{ being } k \text{ given the sequence of symbols } \mathbf{X}$$

© 2006 Scott Pegg



## Matching a symbol with a state

We start by breaking  $P(\mathbf{X}, \pi_i=k)$  into two parts,

$$P(\mathbf{X}, \pi_i=k) = \underbrace{P(x_1, \dots, x_i, \pi_i=k)}_{\text{front}} \underbrace{P(x_{i+1}, \dots, x_L | \pi_i=k)}_{\text{back}}$$

$$= f_k(i) \circ b_k(i)$$

$$P(\pi_i=k | \mathbf{X}) = \frac{P(\mathbf{X}, \pi_i=k)}{P(\mathbf{X})} = \frac{f_k(i) b_k(i)}{P(\mathbf{X})}$$

© 2006 Scott Pegg

## Underflow Problems Again

This time, however, we can't just convert to Logs.

$$f_j(i+1) = e_j(x_{i+1}) \sum_{k \in |S|} f_k(i) a_{kj}$$

Instead we scale the values,

$$\tilde{f}_j(i) = \frac{f_j(i)}{\prod_{n=1}^i s_n}$$

© 2006 Scott Pegg

## Scaling

The iteration equations become

$$\begin{aligned}\tilde{f}_j(i+1) &= \frac{1}{s_{i+1}} e_k(x_{i+1}) \sum_{k \text{ in } |S|} \tilde{f}_k(i) a_{kj} \\ \tilde{b}_j(i+1) &= \frac{1}{s_i} \sum_{k \text{ in } |S|} a_{kj} \tilde{b}_k(i) e_k(x_{i+1})\end{aligned}$$

How do we choose  $s_i$ ?

$$\begin{aligned}\text{Such that } \sum_j f_j(i) &= 1 \\ \text{so } s_{i+1} &= \sum_j e_j(x_{i+1}) \sum_k f_k(i) a_{kj}\end{aligned}$$

© 2006 Scott Pegg

## The 'likelihood' answer

So what does answering the 'likelihood' question give us?

It tells us how well our observed sequence fits our model.

When the HMM is trained on a set of homologous sequences, the likelihood is a measure of whether our new sequence belongs to this family of sequences.

So how do we train an HMM on a set of sequences?

© 2006 Scott Pegg

## The 'learning' question

How should the transition and emission probabilities be updated given new sequences of symbols?

Given  $n$  sequences  $\{\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \dots, \mathbf{X}^{(n)}\}$   
of lengths  $L^{(1)}, L^{(2)}, \dots, L^{(n)}$

which were generated from HMM  $M(\Psi, S, \theta)$ .

Assign values to  $\theta$  that maximize the probabilities of our sequences given the model.

© 2006 Scott Pegg

Since the sequences are assumed to have been generated independently,

$$P(\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(n)} | \theta) = \prod_{i=1}^n P(\mathbf{X}^{(i)} | \theta)$$

We're multiplying small numbers again...

$$\begin{aligned} \text{Score}(\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(n)} | \theta) &= \text{Log } P(\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(n)} | \theta) \\ &= \sum_{i=1}^n \text{Log } P(\mathbf{X}^{(i)} | \theta) \end{aligned}$$

Our goal is to find  $\theta^*$  such that

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \{ \text{Score}(\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(n)} | \theta) \}$$

© 2006 Scott Pegg

## Maximum Likelihood Estimators

Say we know the state sequences  $\mathbf{P}^{(1)}, \dots, \mathbf{P}^{(n)}$   
for each sequence of symbols  $\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(n)}$ .

We can simply count

$A_{kj}$  = # of transitions from state  $k$  to state  $j$

$E_k(b)$  = # of times symbol  $b$  was emitted from  
state  $k$

Our Maximum Likelihood Estimators are then

$$a_{kj} = \frac{A_{kj}}{\sum_{q \in S} A_{kq}} \quad e_k(b) = \frac{E_k(b)}{\sum_{\sigma \in \Psi} E_k(\sigma)}$$

© 2006 Scott Pegg

## Maximum Likelihood Estimators

But there's a potential problem here...

$$a_{kj} = \frac{A_{kj}}{\sum_{q \in S} A_{kq}} \quad e_k(b) = \frac{E_k(b)}{\sum_{\sigma \in \Psi} E_k(\sigma)}$$

possible zero denominators (especially when the  
number of sequences is small).

Laplace Correction

$$\begin{aligned} A'_{kj} &= A_{kj} + r_{kj} \\ E'_k(b) &= E_k(b) + r_k(b) \end{aligned}$$

Where  $r_{kj}$  and  $r_k(b) = 1$   
(or contain a priori knowledge)

© 2006 Scott Pegg

## The Baum-Welch Algorithm

What if we don't know the state sequences?

In this case, finding the optimal parameter values ( $\theta^*$ ) is NP-complete

But we can use an iterative algorithm to get close...

Step 1: Assign initial values to  $\theta$

© 2006 Scott Pegg

## The Baum-Welch Algorithm

Step 2a: Compute the expected number of transitions from state  $k$  to state  $j$

$$P(\pi_i=k, \pi_{i+1}=j \mid \mathbf{X}, \theta) = \frac{f_k(i) a_{kj} e_j(x_{i+1}) b_j(i+1)}{P(\mathbf{X})}$$

so the expected value is

$$A_{kj} = \sum_{h=1}^n \left[ \frac{1}{P(\mathbf{X}^{(h)})} \sum_{i=1}^{L^{(h)}} f_k^{(h)}(i) a_{kj} e_j(x_{i+1}^{(h)}) b_j^{(h)}(i+1) \right]$$

$n$  = number of sequences in  $\mathbf{X}$

$L$  = length of sequence  $\mathbf{X}^{(h)}$

## The Baum-Welch Algorithm

Step 2b: Compute the expected number of emissions of symbol  $b$  from state  $k$

$$E_k(b) = \sum_{h=1}^n \left[ \frac{1}{P(X^{(h)})} \sum_{i | X_i^{(h)} = b} f_k^{(h)}(i) b_k^{(h)}(i) \right]$$

Step 3: Recalculate  $a_{kj}$  and  $e_k(b)$  using the values of  $A_{kj}$  and  $E_k(b)$  using the maximum likelihood estimators.

© 2006 Scott Pegg

## The Baum-Welch Algorithm

Step 4: Calculate

$$\text{Score}(\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(n)} | \boldsymbol{\theta}) = \sum_{i=1}^n \text{Log } P(\mathbf{X}^{(i)} | \boldsymbol{\theta})$$

If the improvement is less than some threshold  $t$ , then stop. Else, go back to step 2.

We're guaranteed to converge, since the function is monotonically increasing and the logs of probabilities are bounded by zero.

There's no guarantee, however, of finding the global maximum, so in general you repeat several times with different initial values of  $\boldsymbol{\theta}$ .

© 2006 Scott Pegg

## Choosing initial parameter values

G	T	G	A	A	G
G	T	C	A	A	G
G	C	C	T	T	G
G	T	-	A	A	G
C	T	-	A	T	G
C	T	-	A	A	G

How do I account for unobserved symbols?

© 2006 Scott Pegg

## Choosing initial parameter values

We can add a pseudocount

Laplace  
Correction

$$A'_{kj} = A_{kj} + r_{kj}$$

Or we can assume there's a background probability for each symbol.

Where can we get these probabilities?

Substitution matrix

This implies that all columns of the alignment come from the same distribution.

© 2006 Scott Pegg

G	T	G	A	A	G
G	T	C	A	A	G
G	C	C	T	T	G
G	T	-	A	A	G
C	T	-	A	T	G
C	T	-	A	A	G

A better method is to consider each column as having been generated from a distribution of symbols

### Dirichlet mixtures

Brown, Hughey, Krogh, Mian, Sjolander, & Haussler (1993) "Using Dirichlet Mixture Priors to Derive Hidden Markov Models for Protein Families", ISMB93  
<http://citeseer.ist.psu.edu/brown93using.html>

© 2006 Scott Pegg

$$p_i = P(\text{amino acid type } i)$$

$\vec{p}$  = probability distribution over amino acid alphabet

$$\sum_i p_i = 1$$

$$\rho(\vec{p}) = \frac{\prod_i p_i^{\alpha_i - 1}}{Z} \quad \text{Dirichlet density}$$

where  $\alpha_i > 0$

$Z$  = normalizing constant (making  $\rho$  sum to 1)

© 2006 Scott Pegg



$$\rho = q_1\rho_1 + \dots + q_n\rho_n$$

$$\sum_i q_i = 1$$

Dirichlet mixture

Typically, a 9-component mixture is used (determined empirically).

Note that the pseudocount method is essentially a Dirichlet mixture with only a single component.

Deriving the parameters and probabilities from alignment data is pretty involved. For details, see

*Sjolander, K., Karplus, K., Brown, M., Hughey, R., Krogh, A., Mian, I.S., and Haussler, D. Dirichlet Mixtures: A Method for Improving Detection of Weak but Significant Protein Sequence Homology. CABIOS, 12(4): 327-345, Aug 1996.*

© 2006 Scott Pegg

## HMM architecture

How do we choose the state architecture in the first place?

Use what's intuitive and what's worked

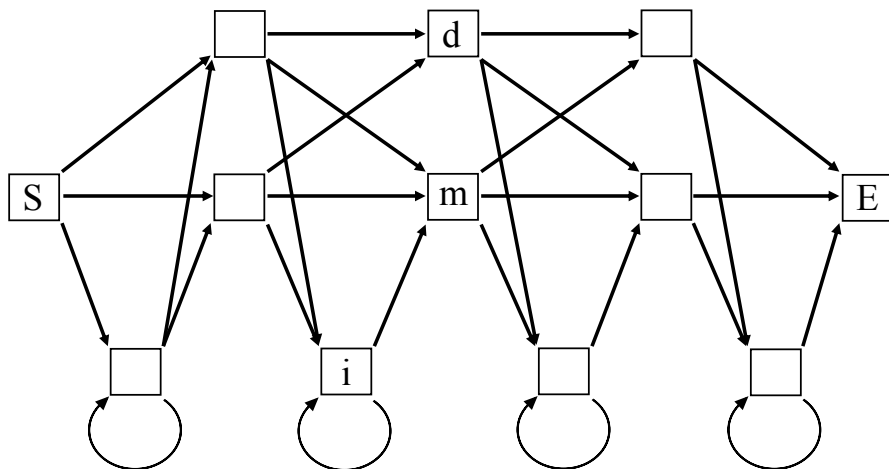
Algorithms to 'learn' the architecture exist

Slow

Unclear if they find an optimal structure

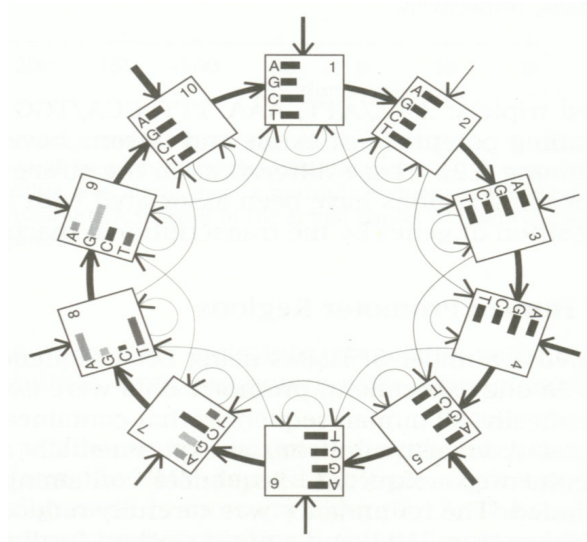
© 2006 Scott Pegg

## A typical bioinformatics HMM



© 2006 Scott Pegg

## HMM architecture variants



© 2006 Scott Pegg

## HMMs in Bioinformatics

- Sequence classification
- Remote homology detection
- Multiple sequence alignment
- General pattern recognition

© 2006 Scott Pegg

## Benefits & Limitations

What are the benefits to using HMMs?

- Solid basis in probability theory
- Relatively fast algorithms
- Multiple uses
- Can be made modular

What are the limitations when using HMMs?

- Need examples to train the model
- First order approximation may miss long-range interactions

© 2006 Scott Pegg

## Some HMM Packages

HMMR & Pfam - Sean Eddy (Sanger Centre)

HMMR is an HMM package

Pfam is a curated database of HMMs trained on protein domains

SAM - David Haussler (U.C. Santa Cruz)

SAM = Sequence Alignment and Modelling System

HMMpro - Baldi & Chauvin (NetID, Inc.)

© 2006 Scott Pegg

## Takeaway

- Basic Probability Theory
- Markov Chains
- Hidden Markov Models
- 3 basic questions of HMMs
  - Decoding - Viterbi
  - Likelihood - forward & backward
  - Learning - Baum-Welch
- Uses, costs & benefits of using HMMs

© 2006 Scott Pegg

## Some HMM References

Rabiner, L.R. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. IEEE*, Vol. 77, pp. 257-286, 1989

Krogh, A., M. Brown, I.S. Mian, K. Sjolander, and D. Haussler. "Hidden Markov Models in Computational Biology: Applications to Protein Modeling," *J. Mol. Biol.*, Vol. 235, pp.1501-1531, 1994

Durbin, R., S. Eddy, A. Krogh, G. Mitchison. *Biological Sequence Analysis*. Cambridge U. Press, 1998

Baldi & Brunak, *Bioinformatics: The Machine Learning Approach*, MIT Press, 1998