CSE 527
Lecture 17
Markov Models and Hidden Markov Models
Markov & Hidden Markov Models

Reference: Durbin Eddy Krogh Mitchison
Biological Sequence Analysis
Cambridge '98

A Key Issue:
So far all sequence models assume independence of different positions - unrealistic.
Example: "CpG Islands"

- CpG - adjacent on one strand, not Watson-Crick pair
- C of CpG often "methylated"
- methyl-C often mutates to T
- CpG less common than expected
  \[ \text{freq}(\text{CpG}) < \text{freq}(C) \cdot \text{freq}(G) \]
- But gene promoter regions usually unmethylated, so CpG \(\rightarrow\) TpG not happening there: "CpG island"
CpG Islands

- More CpG than elsewhere
- More C & G
- Typical length: few 100–few thousand bases

Questions

- Given short sequence (say 100 bp)
  Is it CpG Island or not?
- Given long sequence (say 10–100 kbp)
  Find CpG islands in it.
Markov Chain

A sequence of random variables $X_1, X_2, \ldots$ is a $k$-th order Markov chain if:

$$P_Y(X_i | X_1, X_2, \ldots, X_{i-1}) = P_Y(X_i | X_{i-k}, X_{i-k+1}, \ldots, X_{i-1})$$

i.e. $i$th value is independent of all but previous $k$ values.

**Example 1**
uniform random $A, C, T, G$.

**Example 2**
Weight Matrix Model

**Example 3**
$A, C, G, T$, but $P_Y(C | G$ following $C) =$ lower.
States: \( A, C, G, T \)

Emission: corresponding letter

Transition: \( q_{st} = P(X_i = t | X_{i-1} = s) \)
States: A, C, G, T

Emission: corresponding letter

Transition: $q_{st} = P(X_i = t | X_{i-1} = s)$

Begin / End States
Probability of Emulating Sequence $x$

$$x = x_1 \ x_2 \ \ldots \ x_n$$

$$P(x) = P(x_1, x_2, \ldots, x_n)$$

$$= P(x_n | x_{n-1} x_{n-2} \ldots x_1) \cdot P(x_{n-1} | x_{n-2} \ldots x_1) \ldots P(x_1)$$

$$= P(x_n | x_{n-1}) \cdot P(x_{n-1} | x_{n-2}) \ldots P(x_2 | x_1) \cdot P(x_1)$$

$$= P(x_1) \prod_{i=1}^{n-1} a_{x_i, x_{i+1}}$$
Training

MLE's for transition probabilities are frequencies of transitions when emitting training sequences.
and derived two Markov chain models, one for the regions labelled as CpG islands (the ‘+’ model) and the other from the remainder of the sequence (the ‘−’ model). The transition probabilities for each model were set using the equation

$$a_{st}^+ = \frac{c_{st}^+}{\sum_{s'}c_{st}^+},$$

(3.3)

and its analogue for $a_{st}^−$, where $c_{st}^+$ is the number of times letter $t$ followed letter $s$ in the labelled regions. These are the maximum likelihood (ML) estimators for the transition probabilities, as described in Chapter 1.

(In this case there were almost 60 000 nucleotides, and ML estimators are adequate. If the number of counts of each type had been small, then a Bayesian estimation process would have been more appropriate, as discussed in Chapter 11 and below for HMMs.) The resulting tables are

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>0.180</td>
<td>0.274</td>
<td>0.426</td>
<td>0.120</td>
</tr>
<tr>
<td></td>
<td>0.171</td>
<td>0.368</td>
<td>0.274</td>
<td>0.188</td>
</tr>
<tr>
<td></td>
<td>0.161</td>
<td>0.339</td>
<td>0.375</td>
<td>0.125</td>
</tr>
<tr>
<td></td>
<td>0.079</td>
<td>0.355</td>
<td>0.384</td>
<td>0.182</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>−</td>
<td>0.300</td>
<td>0.205</td>
<td>0.285</td>
<td>0.210</td>
</tr>
<tr>
<td></td>
<td>0.322</td>
<td>0.298</td>
<td>0.078</td>
<td>0.302</td>
</tr>
<tr>
<td></td>
<td>0.248</td>
<td>0.246</td>
<td>0.298</td>
<td>0.208</td>
</tr>
<tr>
<td></td>
<td>0.177</td>
<td>0.239</td>
<td>0.292</td>
<td>0.292</td>
</tr>
</tbody>
</table>

where the first row in each case contains the frequencies with which an A is followed by each of the four bases, and so on for the other rows, so each row
Discrimination/Classification

Calculate log likelihood ratio for (pG model vs background model)

\[ S(x) = \log \frac{P(x \mid +\text{model})}{P(x \mid -\text{model})} = \sum_{i=1}^{n} \log \frac{a_{x_{i-1},x_{i}}^{+}}{a_{x_{i-1},x_{i}}^{-}} \]
les the probability for G following C is lower than that for C following G, the effect is stronger in the ‘−’ table, as expected.

e these models for discrimination, we calculate the log-odds ratio

\[ S(x) = \log \frac{P(x|\text{model} +)}{P(x|\text{model} -)} = \sum_{i=1}^{L} \log \frac{a_{x_{i-1}x_i}^+}{a_{x_{i-1}x_i}^-} \]

\[ = \sum_{i=1}^{L} \beta_{x_{i-1}x_i} \]

is the sequence and \( \beta_{x_{i-1}x_i} \) are the log likelihood ratios of corresponding probabilities. A table for \( \beta \) is given below in bits:\(^1\)

<table>
<thead>
<tr>
<th>( \beta )</th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>−0.740</td>
<td>0.419</td>
<td>0.580</td>
<td>−0.803</td>
</tr>
<tr>
<td>C</td>
<td>−0.913</td>
<td>0.302</td>
<td>1.812</td>
<td>−0.685</td>
</tr>
<tr>
<td>G</td>
<td>−0.624</td>
<td>0.461</td>
<td>0.331</td>
<td>−0.730</td>
</tr>
<tr>
<td>T</td>
<td>−1.169</td>
<td>0.573</td>
<td>0.393</td>
<td>−0.679</td>
</tr>
</tbody>
</table>

\(^3\) shows the distribution of scores, \( S(x) \), normalised by dividing by length, i.e. as an average number of bits per molecule. If we had not nor-

by length, the distribution would have been much more spread out.
Figure 3.2 The histogram of the length-normalised scores for all the sequences. CpG islands are shown with dark grey and non-CpG with light grey.
Above answers Q1: "given short sequence, is it more likely to be from feature model or background model?"

What about Q2: "given long sequence, where are features in it?"

Approach 1: Score, say, 100 base windows.
Pro: simple
Con: arbitrary window; fixed

Approach 2: Combine + 2 - models
Hidden Markov Models (HMMs)

States: 1, 2, ...

Paths: sequence of states \( \pi = (\pi_1, \pi_2, ..., \pi_n) \)

Transitions: \( a_{kl} = \text{Prob}(\pi_i = l \mid \pi_{i-1} = k) \)

Emissions: \( e_k(b) = \text{Prob}(x_i = b \mid \pi_i = k) \)

Observed Data: only emission seq.

Hidden Data: The state/transition seq.
Example: "The occasionally dishonest casino"

1 fair die
1 loaded die
occasionally swap them

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>12</td>
</tr>
</tbody>
</table>
Figure 3.5 The numbers show 300 rolls of a die as described in the example. Below is shown which die was actually used for that roll (F for fair and L for loaded). Under that the prediction by the Viterbi algorithm is shown.

the model as described earlier. Each roll was generated either with the fair die (F) or the loaded one (L), as shown below the outcome of the roll in Figure 3.5. The Viterbi algorithm was used to predict the state sequence, i.e. which die was used for each of the rolls. Generally, as you can see, the Viterbi algorithm has recovered the state sequence fairly well. ☐

Exercise
Joint probability of given path $\Pi$, seq $x$

$$P(x, \Pi) = a_{0 \Pi_1} \prod_{i=1}^{n} c_{\Pi_i} (x_i) a_{\Pi_i \Pi_{i+1}}$$

But $\Pi_i$ hidden

Alternatives

1. Most probable (single) path $\hat{\Pi} = \arg \max_{\Pi} P(x, \Pi)$
2. Sequence of most probable states
   $$\hat{\pi}_i = \arg \max_{K} (P(C \Pi_i = K \mid x))$$
3. ...
The Viterbi Algorithm: Most Probable Path

\[ \Pi^* = \text{argmax}_{\Pi} P(x, \Pi) \]

- Often true that 1 path dominates all others (if not, other approaches may be preferable)
- Key Problem: exponentially many \( \Pi \)
Viterbi

\[ V_k(i) = \text{Probability of most probable path ending in state } k \text{ after emitting } x_1 \ldots x_i \]

\[ V_k(i+1) = e_k(x_{i+1}) \max_k (V_k(i)e_{k_2}) \]

Initialize:
\[ V_k(0) = \begin{cases} 1 & \text{if } k \text{ = start state} \\ 0 & \text{otherwise} \end{cases} \]
Viterbi Traceback

- Above finds \textit{probability} of best path
- To find the path itself, trace backward to state $k$ attaining the max at each stage
More on HMMs:
- Viterbi, forward, backward
- Posterior decoding
- Training: Viterbi & Baum-Welch
- Model structure
Viterbi: \[
\max_{\pi} P(X, \pi)
\]

Most probable path thru A
but B is most probable start at step 2.
Viterbi: Best path to each node

Sequence positions →

\[ V_k(i+1) = \varepsilon_k(x_{i+1}) \cdot \max_k \left( V_k(i) \cdot a_{k} \right) \]
For each state/time want total prob. of all paths leading to it:

\[ f_k(i) = P(X_1, \ldots, X_i \mid \pi_i = k) \]

\[ f_{l+1}(i+1) = e_l(X_{i+1}) \sum_k f_k(i) a_{kl} \]

\[ P(X) = \sum_k f_k(\pi) a_{k0} \]
**Backward Algorithm**

\[ b_k(i) = P(X_{i+1} \ldots X_n | \pi_i = k) \]

\[ b_k(i) = \sum_{\ell} a_{k\ell} e_{\ell}(X_{i+1}) b_{\ell}(i+1) \]

\[ b_k(n) = a_{k0} \]
\[ P(X_i, \pi_i = k) = P(X_1 \ldots X_i, \pi_i = k) \frac{P(X_{i+1} \ldots X_n | \pi_i = k)}{P(X_i \ldots X_n | \pi_i = k)} = \frac{f_k(i)}{b_k(L_i)} \frac{P(X_i \ldots X_n | \pi_i = k)}{P(X_i \ldots X_n | \pi_i = k)} \]
Posterior Decoding, I

\[ \hat{\pi}_i = \arg\max_k \left( P(\Theta_i = k | x) \right) \]
Example: "The occasionally dishonest casino"

1 fair die
1 loaded die
occasionally swap them

| 1: 1/6 |
| 2: 1/6 |
| 3: 1/6 |
| 4: 1/6 |
| 5: 1/6 |
| 6: 1/6 |

| 1: 4/10 |
| 2: 1/10 |
| 3: 1/10 |
| 4: 1/10 |
| 5: 1/10 |
| 6: 1/10 |
| 7: 1/10 |

.95 → .05
.05 → .01
3.2 Hidden Markov models

Rolls 3151162464466442453113216311641521336251445436316566265666666
Die FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLL
Viterbi FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLL

Rolls 651166453132651245636664631636663162326455236266666666666666
Die LLLLLLLLLLLLLLLLLLLLLLLLLLFFL
Viterbi LLLLLLLLLLLLLLLLLLLLLLLLLLFFL

Rolls 222555441666566563564324364131513465146353411126414626253356
Die FFFFFFFFFLLLLLL
Viterbi FFFFFFFFFLLLLLL

Rolls 36616366466232534413661661632525624622525265252266435353336
Die LLLLLLLLLLLLLL
Viterbi LLLLLLLLLLLLLL

Rolls 233121625364414432335163243633665562466662632666612355245242
Die FFFFFFFFFFFFFFFFFFFFFFFFFL
Viterbi FFFFFFFFFFFFFFFFFFFFFFFFFL

Figure 3.5 The numbers show 300 rolls of a die as described in the example. Below is shown which die was actually used for that roll (F for fair and L for loaded). Under that the prediction by the Viterbi algorithm is shown.

the model as described earlier. Each roll was generated either with the fair die (F) or the loaded one (L), as shown below the outcome of the roll in Figure 3.5. The Viterbi algorithm was used to predict the state sequence, i.e. which die was used for each of the rolls. Generally, as you can see, the Viterbi algorithm has recovered the state sequence fairly well.

Exercise
Figure 3.6 The posterior probability of being in the state corresponding to the fair die in the casino example. The x axis shows the number of the roll. The shaded areas show when the roll was generated by the loaded die.

The first approach is to define a state sequence \( \hat{\pi}_i \) that can be used in place of \( \pi_i^* \),

\[
\hat{\pi}_i = \arg\max_k P(\pi_i = k| x). \tag{3.15}
\]

As suggested by its definition, this state sequence may be more appropriate when we are interested in the state assignment at a particular point \( i \), rather than the complete path. In fact, the state sequence defined by \( \hat{\pi}_i \) may not be particularly likely as a path through the entire model; it may even not be a legitimate path at all if some transitions are not permitted, which is normally the case.

The second, and perhaps more important, new decoding approach arises when it is not the state sequence itself which is of interest, but some other property derived from it. Assume we have a function \( g(k) \) defined on the states. The natural value to look at then is

\[
G(i|x) = \sum P(\pi_i = k| x) g(k). \tag{3.16}
\]
Posterior Decoding, II

\[ \hat{\pi}_i = \text{arg}\max_k \left( P(\pi_i = k | x) \right) \]

\[ g(k) \text{ function on states} \]

\[ g(i | x) = \sum_k P(\pi_i = k | x) \cdot g(k) \]
CpG Islands Again

Data: 41 human seqs, totaling 60kbp, W/ 48 CpG islands
avg length ~ 1kbp each

Viterbi
Found 46 of 48
Plus 121 "false pos"

Posterior decoding
Same 2 false neg
236 false pos

Post Process:
marge within 500
distance < 500

46/48
Plus 67 false pos

again 46/48
Plus 83 false neg
Training

Given model topology
Given \( t \) independent training sequences
Want to learn transition & emission probabilities

If \( \pi \) known, then

MLE \( a_{k; \ell} = \frac{\text{Count } K \rightarrow \ell}{\text{Count } K \rightarrow \text{any other}} \)

\( E_k(b) = \) similar

\( \pi \) hidden
Use EM: given \( \pi \) can estimate \( \Theta \)
Viterbi training

- Make initial parameter estimates
- Calca Viterbi path for each training sequence
- Count transitions & emissions → new $\Theta$
- Iterate

- Not rigorously optimizing derived likelihood.
  (But still useful)
Baum-Welch Training

\[ P(\pi_i \rightarrow k, \pi_{i+1} = \ell | \chi, \Theta) = \frac{f_k(i) \cdot \phi_{\ell}(x_{i+1}) \cdot b_{\ell}(\ell+1)}{P(\chi \Theta)} \]

\[ E(\#T_{K\rightarrow \ell}) = \sum_{\text{training series}} \frac{1}{P(\chi_{x_j})} \sum_{i} \]

Emissions model
Example: The occasionally dishonest casino, part 5

We are suspicious that a casino is operated as described in the example on p. 54, but we do not know for certain. Night after night we collect data by simply observing rolls. When we have enough, we want to estimate a model. Assume the data we collected were the 300 rolls shown in Figure 3.5. From this sequence of observations a model was estimated by the Baum–Welch algorithm. Initially all the probabilities were set to random numbers. Here are diagrams of the model that generated the data (identical to the one in the example on p. 54) and the estimated model.

You can see they are fairly similar, although the estimated transition probabilities are quite different from the real ones. This is partly a problem of local minima, and by trying more times it is actually possible to obtain a model closer to the correct one. However, from a limited amount of data it is never possible to estimate the parameters exactly.

To illustrate the last point, 30 000 random rolls were generated (data are not
shown!), and a model was estimated. This came very close to the correct one:

![Diagram showing transitions between states: Fair to Fair with probability 0.93, Fair to Loaded with probability 0.07, Loaded to Fair with probability 0.12, Loaded to Loaded with probability 0.88.]

To see how good these models are compared to just assuming a fair die all the time, the log-odds per roll was calculated using the 300 observations for the three models:

- The correct model: 0.101 bits
- Model estimated from 300 rolls: 0.097 bits
- Model estimated from 30000 rolls: 0.100 bits

The worst model estimated from 300 rolls has almost the same log-odds as the two other models. That is because it is being tested on the same data as it was estimated from. Testing it on an independent set of rolls yields significantly lower log-odds than the other two models.

Exercises
3.5 Derive the result (3.19). Use the fact that

\[ P(\pi_i = k, \pi_{i+1} = l|x, \theta) = \frac{1}{P(x|\theta)} P(x, \pi_i = k, \pi_{i+1} = l|\theta), \]
Summary

Viterbi
- best single path
- Max of product
- Forward
- Summary over all paths
- Sum of products
- Backward
- Similar

Baum Welch
- Training based on EM & F/B
Model Structure

Define structure as well as you can.

\[ p \rightarrow p^n(1-p) \]
TALKS

Today 3:30 MEB 243 (CSE590CB)
Covariance models for finding non-coding RNA

Wednesday 3:30 Hitchcock 132 (G5 Seminar)
"Small non-coding RNA's & Animal Development"

Monday 12/8 3:30 MEB 243 (CSE590CB)
Speeding up covariance models

Wednesday 12/3 K-069 (Combi)
Me: "Improved Gene Selection Using Microarray"
HMM's in Action: pfam

- Proteins fall into families, both across & within species
  Ex: Globins, GPCRs, ...

- Identifying family is very useful - suggests function, etc.

- So search & alignment are important

- One successful approach profile HMM's
<table>
<thead>
<tr>
<th>Helix</th>
<th>HBA_HUMAN</th>
<th>HBB_HUMAN</th>
<th>MYG_PHYCA</th>
<th>GLB3_CHITP</th>
<th>GLB5_PETMA</th>
<th>LGB2_LUPLU</th>
<th>GLB1_GLYDI</th>
<th>Consensus</th>
</tr>
</thead>
<tbody>
<tr>
<td>--------------</td>
<td>---------------</td>
<td>---------------</td>
<td>--------------</td>
<td>--------------</td>
<td>--------------</td>
<td>--------------</td>
<td>--------------</td>
<td>-----------</td>
</tr>
<tr>
<td>AAAA...AAAAA</td>
<td>BBBBBB...BBBBB</td>
<td>CCCCCC...CCCCC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Helix**

<table>
<thead>
<tr>
<th>Helix</th>
<th>HBA_HUMAN</th>
<th>HBB_HUMAN</th>
<th>MYG_PHYCA</th>
<th>GLB3_CHITP</th>
<th>GLB5_PETMA</th>
<th>LGB2_LUPLU</th>
<th>GLB1_GLYDI</th>
<th>Consensus</th>
</tr>
</thead>
<tbody>
<tr>
<td>HBA_HUMAN</td>
<td>-DLS-----</td>
<td>-HGSQ...</td>
<td>-GDLSTP...</td>
<td>KHLKT...</td>
<td>AG...</td>
<td>LG------</td>
<td>SG-----</td>
<td>t</td>
</tr>
<tr>
<td>HBB_HUMAN</td>
<td>-HGSAQV...</td>
<td>-HGSQV...</td>
<td>-GDLSTP...</td>
<td>KHLKT...</td>
<td>AG...</td>
<td>LG------</td>
<td>SG-----</td>
<td>t</td>
</tr>
<tr>
<td>MYG_PHYCA</td>
<td>-HGSAQV...</td>
<td>-HGSQV...</td>
<td>-GDLSTP...</td>
<td>KHLKT...</td>
<td>AG...</td>
<td>LG------</td>
<td>SG-----</td>
<td>t</td>
</tr>
<tr>
<td>GLB3_CHITP</td>
<td>-HGSAQV...</td>
<td>-HGSQV...</td>
<td>-GDLSTP...</td>
<td>KHLKT...</td>
<td>AG...</td>
<td>LG------</td>
<td>SG-----</td>
<td>t</td>
</tr>
<tr>
<td>GLB5_PETMA</td>
<td>-HGSAQV...</td>
<td>-HGSQV...</td>
<td>-GDLSTP...</td>
<td>KHLKT...</td>
<td>AG...</td>
<td>LG------</td>
<td>SG-----</td>
<td>t</td>
</tr>
<tr>
<td>LGB2_LUPLU</td>
<td>-HGSAQV...</td>
<td>-HGSQV...</td>
<td>-GDLSTP...</td>
<td>KHLKT...</td>
<td>AG...</td>
<td>LG------</td>
<td>SG-----</td>
<td>t</td>
</tr>
<tr>
<td>GLB1_GLYDI</td>
<td>-HGSAQV...</td>
<td>-HGSQV...</td>
<td>-GDLSTP...</td>
<td>KHLKT...</td>
<td>AG...</td>
<td>LG------</td>
<td>SG-----</td>
<td>t</td>
</tr>
<tr>
<td>Consensus</td>
<td>LS...vawkv</td>
<td>g...gfl...</td>
<td>FG</td>
<td>FG</td>
<td>FG</td>
<td>FG</td>
<td>FG</td>
<td>FG</td>
</tr>
</tbody>
</table>

### Figure 5.1
An alignment of seven globins from Bashford, Chothia & Lesk [1987]. To the left is the protein identifier in the SWISS-PROT database [Bairoch & Apweiler 1997]. The eight alpha helices are shown as A–H above the alignment. A consensus line below the alignment indicates residues that are identical among at least six of the seven sequences in uppercase, ones identical in four or five sequences in lowercase, and positions where there is a residue identical in three sequences with a dot.
Profile HMM Structure

**Figure 5.2** The transition structure of a profile HMM. We use diamonds to indicate the insert states and circles for the delete states.

- **Mj:** Match states (20 emission probabilities)
- **lj:** Insert states (Background emission probabilities)
- **Dj:** Delete states (silent - no emission)
How Profile HMM used

- Search
  Forward or Viterbi algorithm
  Scoring - log likelihood (length adjusted) log odds vs background

- Alignment
  Viterbi

See next slide
Figure 5.5 To the left the length-normalized LL score is shown as a function of sequence length. The right plot shows the same for the log-odds score.
Z-Scores

Figure 5.6 The Z-score calculated from the LL scores (left) and the log-odds (right).
Model Building Refinements

- Pseudo counts (count = 0 common in training w/ 20 ARs)

\[ g_i(a) = \frac{C_{i,a} + A \cdot \varphi_a}{\sum_{a} C_{i,a} + A} \]

\[ \varphi_a = \text{background} \]

\[ A \sim 20 \]

\[ (n \sim 50 + \text{training}) \]

- Pseudo count "mixtures"

  eg separate pseudo count vectors for various contexts (hydrophobic region, buried regions ... )

\[ (n \sim 10 - 20 + \text{training}) \]
Refinements (cont.)

- Weighting: May need to down-weight highly similar sequences to reflect sampling bias, phylogenetic info, etc.

- Match - Insert Assignment
  Simple threshold, e.g. ">50% gap = insert"
  May not be optimal

Can use Forward Alg-like dyn. prog.
Method to compute Max a posteriori segment