ML Bioinformatics
SNU Spring 2021

Profile Hidden Markov Models

Prof. Sun Kim

Presenter: Dohoon Lee (TA)
Contact: dohlee.bioinfo@gmail.com

Agenda

- Recap: HMM basics
- Position-specific scoring matrix
- Profile hidden Markov model

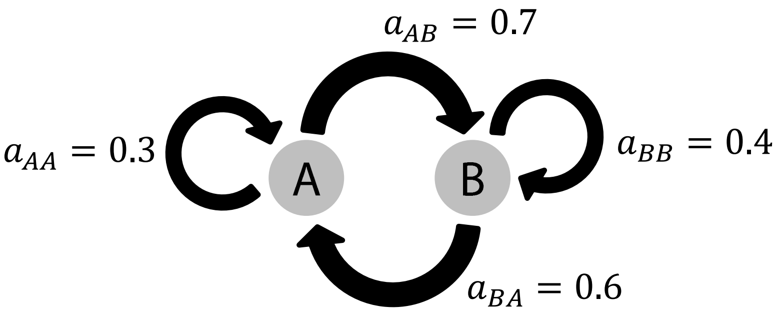
RECAP: HMM BASICS

Recap: MM and HMM

- Markov model (MM)
  - We directly observe a sequence of states, and the transition between states only depends on the recent k states (k-th order MM).
  - Usually, we deal with 1st order MM, where the transition only depends on the current state.

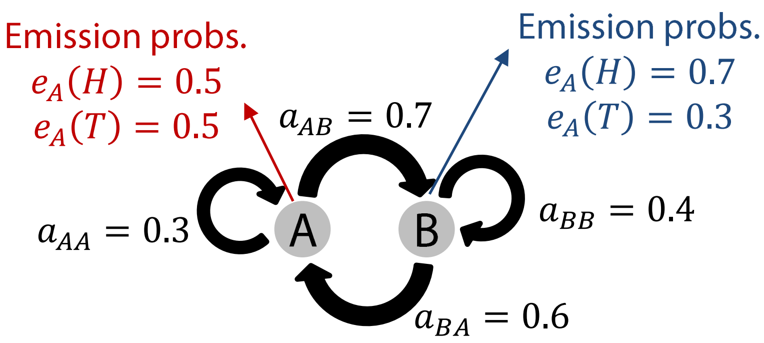
- Hidden Markov model (HMM)
  - We cannot directly observe a sequence of states, instead we observe a sequence of emissions from the hidden states.
  - State transition is Markovian.
  - The problem is, the same observation can be emitted from two or more states. In other words, distinguishing from which hidden state the observation was emitted is not straightforward!

Observation: ABAABBABABAAB



Markov model

Observation: HTHHTHTHTHTHT



Hidden Markov model

Recap: Three questions in HMM

- Evaluation
  - What is a probability of generating an observed sequence x by HMM?
  - Solution: Forward/Backward algorithm

- Decoding
  - Given an observed sequence x, what is the most probable hidden state path of HMM?
  - Solution: Viterbi algorithm

- Learning
  - Given an observed sequence x, estimate model parameters of HMM.
  - Solution: Baum-Welch algorithm

Recap: Viterbi algorithm for decoding

- Given an observed sequence x, what is the most probable hidden state path of HMM?

- Formally: Given an HMM model M, find the hidden state path π that maximizes the probability of observing the observed sequence x.

$$argmax_{\pi}\ P[x \mid M_{HMM}, \pi]$$

- Naïve solution: Compute $P[x|M_{HMM}, \pi]$ for all possible π's.
  - Since the number of possible π's increases exponentially, it is not feasible.

- Better solution: Viterbi algorithm (Dynamic programming formulation)
  - Optimal substructure
    - The optimal path of the prefix of length j that ends with state i must contain one of the optimal path of the prefix of length $j - 1$.
  - Recurrence relation

$$V(i,j) = \max_{k}(\ V(k, j-1) \times a_{ki} \times e_i\left(x_j\right)\ )$$

Recap: Forward/Backward algorithm for evaluation

- What is a probability of generating an observed sequence x by HMM?

- Formally: Given an HMM model M, find the probability $P[x \mid M_{HMM}]$ of observing the observed sequence x.

- Naïve solution: Law of total probability; Weighted sum of all possible $P[O \mid M_{HMM}, \pi]$'s!

$$P[x \mid M_{HMM}] = \sum_{\pi} P[x|M_{HMM}, \pi]P[\pi]$$

- Better solution: Forward/Backward algorithm (Dynamic programming formulation)
  - Optimal substructure
    - The probability of generating an observed prefix of length j with final state as i can be easily computed by utilizing the precomputed probabilities of the prefixes of length $j-1$.
  - Recurrence relation

$$F(i,j) = \sum_{k} F(k, j-1) \times a_{ki} \times e_i(x_j)$$

Recap: Baum-Welch expectation-maximization algorithm for learning

- Given an observed sequence x, estimate model parameters of HMM.
- Formally: Given an observed sequence(s) x and an HMM model, find emission probabilities e and transition probabilities a that maximizes the likelihood $L(e, a|x)$.

- Before dealing with the complex Baum-Welch, let's consider an easy (but unreal) case first — Imagine that hidden state(s) for x is known!

- Then maximum likelihood estimation of $e_k(b)$ (i.e., probability of emitting a character b from state k) becomes

$$e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$$

- Thus, it becomes the relative count of the event of emitting b from state k!
- Similarly, MLE of transition probability a_{kl} is also a relative count:

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}}$$

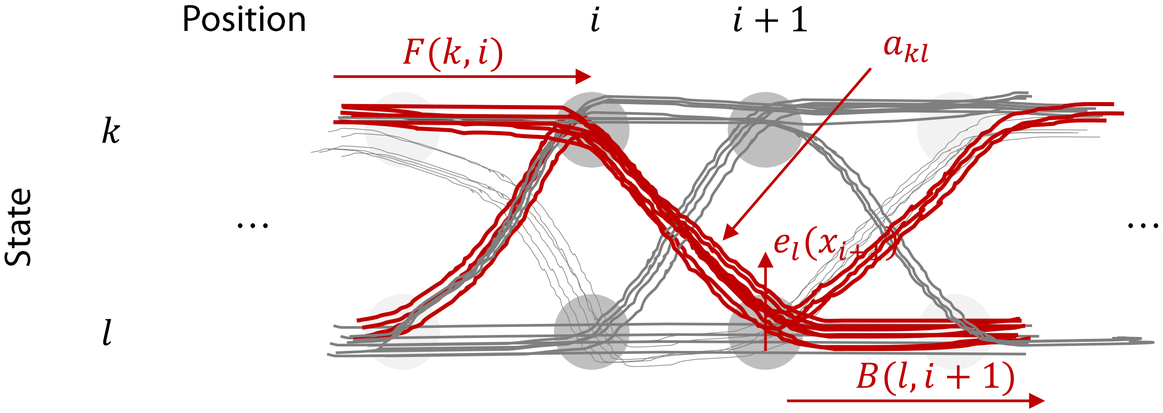
Recap: Baum-Welch expectation-maximization algorithm for learning (cont'd)

- In fact, we do not know the hidden state path.
- However, we can still compute the expected counts of emissions and transitions by manipulating probabilities from forward/backward algorithm!

- First, the expected counts of transition from state k to l, A_{kl} is:

$$A_{kl} = \frac{\sum_i F(k,i) a_{kl} e_l(x_{i+1}) B(l, i+1)}{P(x)}$$

- Visually, $F(k,i)a_{kl}e_l(x_{i+1})B(l,i+1)$ can be depicted as below:



Among numerous possible state paths,
it computes a fraction of state paths that makes transition from state k to state l between position i and $i+1$.

Recap: Baum-Welch expectation-maximization algorithm for learning (cont'd)

- Similarly, the expected counts of the emission of character b from state k, $E_k(b)$ is:

$$E_k(b) = \frac{\sum_{i\ s.t.\ x_i=b} F(k,i)B(k,i)}{P(x)}$$

- After computing the expected counts of A_{kl} and $E_k(b)$, we can now utilize the maximum-likelihood equations we derived by assuming the hidden state path is known.

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}}$$
$$e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$$

- Every iteration of these E/M steps are proven to improve model likelihood.
  - (1) E-step: Computing the expected counts of A_{kl} and $E_k(b)$
  - (2) M-step: Recompute model parameters a and e with maximum likelihood estimates

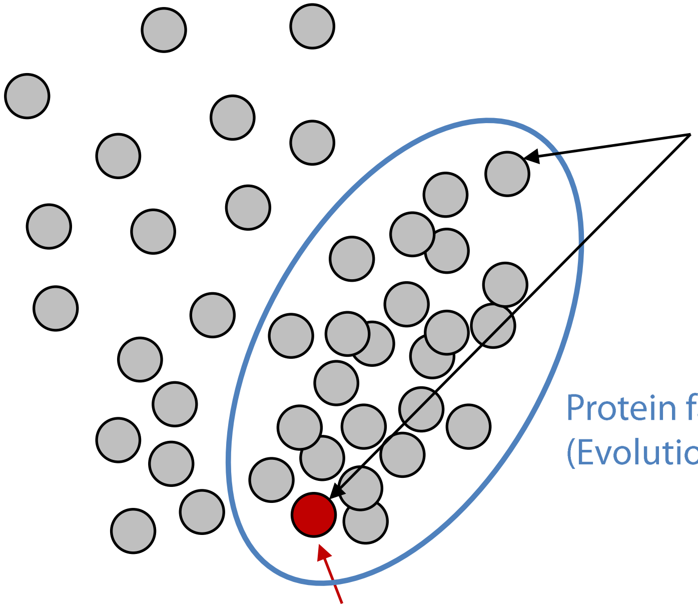
See https://archive.org/details/statisticalanaly00litt/page/n145/mode/2up for the proof of the correctness of EM.

Position-specific scoring matrix & profile hidden Markov model

Going beyond a pair

- Pairwise alignment is a great algorithm to determine whether a pair of sequences are evolutionarily related.

- But it often doesn't work well with distant, but still related sequences.



For these two sequences, pairwise alignment may not produce a good score, but they are still related!

Protein family
(Evolutionarily related proteins)

Query protein sequence

How can we match distantly-related sequences?

- We don’t want to miss those distant relatives — they may provide valuable information for the function of our query protein.

- Why don’t we focus on the shared features of the protein family, so that evolutionarily irrelevant discrepancies have less impact on the result?

```
HBA_HUMAN    A Q V K G H G K K V A D A L T N A V A H
HBB_HUMAN    P K V K A H G K K V L G A F S D G L A H
MYG_PHYCA    E D L K K H G V T V L T A L G A I L K K
GLB3_CHITP   A P F E T H A N R I V G F F S K I I G E
GLB5_PETMA   A D V R W H A E R I I N A V N D A V A S
LGB2_LUPLU   P E L Q A H A G K V F K L V Y E A A I Q
GLB1_GLYDI   P G V A A L G A K V L A Q I G V A V S H
```

Many H’s here!
High penalty should be imposed
if our query doesn’t have H here.

Relatively less conserved
Our query should be penalized much less
for not having Y here.

Example from Durbin et al.

Profile is a computational summary of a set of sequences

- A computational summary of a set of sequences is called profile.
  - Profile summarizes the evolutionary conservation of amino acids.
  - In other words, it informs us the important positions of the query sequence that we should focus our attention on.

- Today, we will discuss two types of profiles.
  1. Position-specific scoring matrix (PSSM)
  2. Profile hidden Markov model (pHMM)

Position-specific scoring matrix (PSSM)

- Position-specific scoring matrix (PSSM) is a simple, but effective form of sequence profile that can be derived from multiple sequence alignment.
  - You can think of a PSSM as a position-specific version of substitution matrix tailored to the given set of sequences.

- We discuss two forms of PSSM: (1) PSSM with log-odds and (2) PSSM with weighted scores.

Derivation of PSSM with log-odds (Probabilistic PSSM)

- For each position i (i.e., column i) of a length L alignment of N_{seq} sequences, we can 'summarize' the relative frequency of amino acid a as

$$f_{i,a} = \frac{n_{i,a}}{N_{seq}},$$

- where $n_{i,a}$ denotes the observed frequency of amino acid a in column i.

- In short, f is a position-specific relative frequency matrix, where its element $f_{i,a}$ is a position i-specific relative frequency of amino acid a.

Position-specific scoring matrix (PSSM) (cont'd)

- With all the $f_{i,b}$'s, we can derive the probability of observing our query sequence x as:

$$P(x|M) = \prod_{i=1}^{L} f_{i,x_i}$$

- Note that the notation of LHS represents this probability is computed under the assumption that x has evolutionary relationship with the set of sequences.
  - Let us denote this assumption as model M here.

- To convert this probability as a score, we should compute a probability of x given a random model R, $P(x|R)$ and compute the log-odds.
  - R: x does not have evolutionary relationship with other sequences.

$$S(x) = \log \frac{P(x|M)}{P(x|R)} = \sum_{i=1}^{L} \log \frac{f_{i,x_i}}{q_{x_i}}$$

- And thus, the values $m_{i,a} = \log \frac{f_{i,a}}{q_a}$ constitutes the PSSM m.

Q. Can you see any shortcomings of this model?

Position-specific scoring matrix (PSSM) (cont'd)

Gribskov, McLachlan & Eisenberg (1987)

Derivation of PSSM with weighted scores (Non-probabilistic PSSM)

- The most widely used form of PSSM
- Intuitive way to score a query sequence against a set of previously aligned sequences: Average pairwise substitution scores

- Denoting PSSM as m, its element $m_{i,a}$ is given as:

$$m_{i,a} = \sum_{b \in AA} f_{i,b} \times s(a,b) \ ,$$

where AA denotes the set of 20 amino acids.

- Then, the score of the query sequence x is

$$\sum_{i=1}^{L} m_{i,x_i}$$

Q. Why is the score above is equivalent to the average pairwise substitution scores?

Q. Why this formulation of PSSM is not probabilistic?

Q. By the way, what is a "probabilistic" scoring scheme?

PSSM with weighted scores: a worked example

- Let's see how we can build a (non-probabilistic) PSSM from a set of aligned sequences and compute a score of our query sequence against it.

	0								1										2
Position	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0
Query	P	Q	W	K	W	H	G	K	K	V	L	D	A	L	Y	N	A	V	A	H
HBA_HUMAN	A	Q	V	K	G	H	G	K	K	V	A	D	A	L	T	N	A	V	A	H
HBB_HUMAN	P	K	V	K	A	H	G	K	K	V	L	G	A	F	S	D	G	L	A	H
MYG_PHYCA	E	D	L	K	K	H	G	V	T	V	L	T	A	L	G	A	I	L	K	K
GLB3_CHITP	A	P	F	E	T	H	A	N	R	I	V	G	F	F	S	K	I	I	G	E
GLB5_PETMA	A	D	V	R	W	H	A	E	R	I	I	N	A	V	N	D	A	V	A	S
LGB2_LUPLU	P	E	L	Q	A	H	A	G	K	V	F	K	L	V	Y	E	A	A	I	Q
GLB1_GLYDI	P	G	V	A	A	L	G	A	K	V	L	A	Q	I	G	V	A	V	S	H

Let's derive a score within PSSM when the query has amino acid W at this position, 3, with BLOSUM62 scoring matrix.

For example, $m_{3,W} = f_{3,V} \times s(W,V) + f_{3,L} \times s(W,L) + f_{3,F} \times s(W,F) = \frac{4}{7} \times (-3) + \frac{2}{7} \times (-2) + \frac{1}{7} \times 1 = -\frac{15}{7}$

HBA_HUMAN	A	Q	V	K	G	H	G	K	K	V	A	D	A	L	T	N	A	V	A	H
HBB_HUMAN	P	K	V	K	A	H	G	K	K	V	L	G	A	F	S	D	G	L	A	H
MYG_PHYCA	E	D	L	K	K	H	G	V	T	V	L	T	A	L	G	A	I	L	K	K
GLB3_CHITP	A	P	F	E	T	H	A	N	R	I	V	G	F	F	S	K	I	I	G	E
GLB5_PETMA	A	D	V	R	W	H	A	E	R	I	I	N	A	V	N	D	A	V	A	S
LGB2_LUPLU	P	E	L	Q	A	H	A	G	K	V	F	K	L	V	Y	E	A	A	I	Q
GLB1_GLYDI	P	G	V	A	A	L	G	A	K	V	L	A	Q	I	G	V	A	V	S	H

Derivation of PSSM

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
P	2.4	0.0	-2.6	-1.1	-1.6	-2.1	-1.6	-1.4	-1.3	-2.3	-2.7	-1.4	-1.7	-3.0	-1.7	-1.3	-1.7	-2.3	-1.4	-1.4
R	-1.3	-0.7	-2.7	1.6	-1.0	-0.3	-1.6	-0.3	2.4	-3.0	-2.3	-0.9	-1.1	-2.7	-1.3	-0.9	-1.7	-2.4	-1.0	0.3
E	-0.1	1.3	-2.4	1.3	-1.1	-0.4	-1.6	0.3	0.4	-2.3	-2.6	-0.4	-1.1	-2.7	-1.0	1.0	-1.7	-2.3	-1.0	1.1
V	-1.1	-2.4	2.4	-1.9	-1.1	-2.4	-1.7	-1.1	-2.0	3.7	1.3	-2.0	-0.3	1.6	-2.0	-1.3	0.4	2.4	-0.6	-2.4
F	-3.0	-3.1	0.3	-2.9	-1.9	-0.9	-2.6	-2.6	-2.9	-0.7	0.4	-2.7	-0.7	1.4	-1.7	-2.6	-1.6	-0.7	-2.0	-2.0
Q	-0.6	0.7	-2.1	1.4	-1.0	-0.3	-1.6	-0.1	0.7	-2.3	-2.1	-0.7	-0.6	-2.4	-0.9	0.0	-1.7	-2.0	-1.0	1.1
L	-2.1	-3.1	1.7	-2.0	-1.7	-2.0	-2.7	-2.0	-1.9	1.3	2.0	-2.7	-0.3	1.7	-2.4	-2.3	-0.6	1.7	-1.3	-2.6
K	-0.7	0.3	-2.1	2.6	-0.6	-1.1	-1.6	0.9	3.3	-2.3	-2.1	-0.3	-1.1	-2.4	-1.0	0.1	-1.7	-2.0	-0.4	0.6
C	-1.9	-3.1	-1.1	-2.7	-1.3	-2.7	-1.7	-2.4	-2.7	-1.0	-1.0	-2.3	-0.9	-1.3	-2.0	-2.4	-0.7	-0.9	-1.1	-2.9
T	-0.6	-1.1	-0.6	-0.9	0.0	-1.9	-1.1	-0.7	-0.1	-0.3	-0.9	-0.1	-0.6	-1.0	0.1	-0.6	-0.6	-0.4	-0.4	-1.1
D	-1.0	1.6	-3.3	-0.7	-1.9	-1.4	-1.4	-0.7	-1.3	-3.0	-3.3	0.1	-2.1	-3.3	-0.7	1.3	-2.1	-3.1	-1.6	-0.3
Y	-2.4	-2.4	-0.4	-1.9	-1.6	1.6	-2.6	-2.0	-2.0	-1.0	-0.6	-2.4	-1.0	0.1	-1.0	-2.1	-1.9	-1.1	-2.0	-0.1
S	0.0	-0.1	-2.0	0.0	0.1	-1.1	0.4	0.0	-0.1	-2.0	-1.6	0.4	0.0	-2.0	1.1	0.0	0.0	-1.6	0.7	0.1
G	-1.1	-0.6	-3.3	-1.7	0.0	-2.3	3.4	-0.4	-2.0	-3.3	-3.1	1.0	-1.3	-3.4	1.0	-1.3	-0.3	-3.0	0.0	-1.7
W	-3.4	-3.1	-2.1	-2.9	-0.7	-2.0	-2.4	-3.0	-2.9	-3.0	-2.0	-2.9	-2.1	-1.6	-2.0	-3.4	-2.9	-2.7	-2.9	-2.4
N	-1.7	0.0	-3.0	-0.3	-1.4	0.4	-0.9	0.1	0.0	-3.0	-2.9	0.7	-2.0	-3.0	0.9	0.4	-2.0	-2.9	-1.1	0.6
M	-1.6	-2.0	1.1	-1.0	-1.3	-1.4	-2.1	-1.3	-1.0	1.0	1.0	-2.0	-0.3	1.0	-1.7	-1.6	-0.7	1.0	-1.0	-1.4
A	1.1	-1.1	-0.6	-0.3	1.1	-1.9	1.7	-0.1	-0.9	-0.3	-0.3	-0.1	1.7	-1.0	-0.3	-0.6	2.0	0.1	1.6	-1.1
I	-2.1	-3.1	2.3	-2.7	-2.0	-2.3	-2.7	-2.0	-2.7	3.3	1.7	-2.7	-0.7	2.0	-2.4	-1.9	0.0	2.3	-1.1	-2.9
H	-1.7	-1.0	-2.7	-0.7	-1.9	6.4	-2.0	-1.1	-0.9	-3.0	-2.6	-1.3	-1.7	-2.4	-0.7	-1.0	-2.3	-2.9	-1.9	3.1

$m_{3,W} = -\frac{15}{7}$

Note the high score of H here!

PSSM with weighted scores: a worked example (cont'd)

- Scoring query sequence
  - In reality, scoring is done in a sliding-window manner and the best hit is reported.

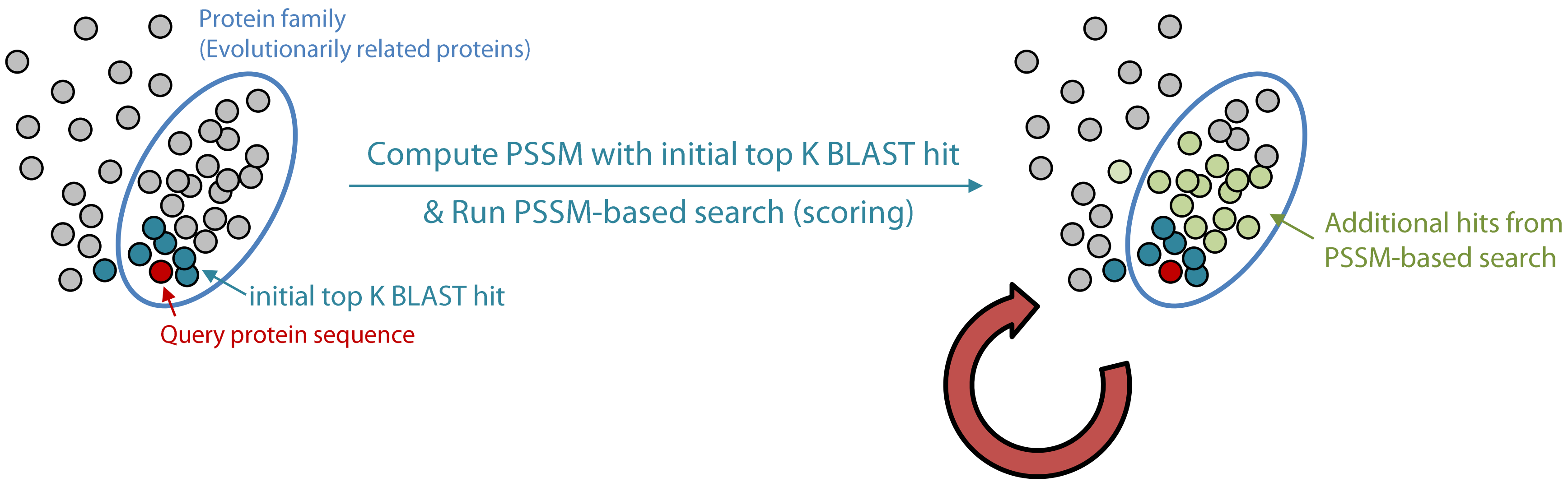
Position	01	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20
Query	P	Q	W	K	W	H	G	K	K	V	L	D	A	L	Y	N	A	V	A	H
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
P	2.4	0.0	-2.6	-1.1	-1.6	-2.1	-1.6	-1.4	-1.3	-2.3	-2.7	-1.4	-1.7	-3.0	-1.7	-1.3	-1.7	-2.3	-1.4	-1.4
R	-1.3	-0.7	-2.7	1.6	-1.0	-0.3	-1.6	-0.3	2.4	-3.0	-2.3	-0.9	-1.1	-2.7	-1.3	-0.9	-1.7	-2.4	-1.0	0.3
E	-0.1	1.3	-2.4	1.3	-1.1	-0.4	-1.6	0.3	0.4	-2.3	-2.6	-0.4	-1.1	-2.7	-1.0	1.0	-1.7	-2.3	-1.0	1.1
V	-1.1	-2.4	2.4	-1.9	-1.1	-2.4	-1.7	-1.1	-2.0	3.7	1.3	-2.0	-0.3	1.6	-2.0	-1.3	0.4	2.4	-0.6	-2.4
F	-3.0	-3.1	0.3	-2.9	-1.9	-0.9	-2.6	-2.6	-2.9	-0.7	0.4	-2.7	-0.7	1.4	-1.7	-2.6	-1.6	-0.7	-2.0	-2.0
Q	-0.6	0.7	-2.1	1.4	-1.0	-0.3	-1.6	-0.1	0.7	-2.3	-2.1	-0.7	-0.6	-2.4	-0.9	0.0	-1.7	-2.0	-1.0	1.1
L	-2.1	-3.1	1.7	-2.0	-1.7	-2.0	-2.7	-2.0	-1.9	1.3	2.0	-2.7	-0.3	1.7	-2.4	-2.3	-0.6	1.7	-1.3	-2.6
K	-0.7	0.3	-2.1	2.6	-0.6	-1.1	-1.6	0.9	3.3	-2.3	-2.1	-0.3	-1.1	-2.4	-1.0	0.1	-1.7	-2.0	-0.4	0.6
C	-1.9	-3.1	-1.1	-2.7	-1.3	-2.7	-1.7	-2.4	-2.7	-1.0	-1.0	-2.3	-0.9	-1.3	-2.0	-2.4	-0.7	-0.9	-1.1	-2.9
T	-0.6	-1.1	-0.6	-0.9	0.0	-1.9	-1.1	-0.7	-0.1	-0.3	-0.9	-0.1	-0.6	-1.0	0.1	-0.6	-0.6	-0.4	-0.4	-1.1
D	-1.0	1.6	-3.3	-0.7	-1.9	-1.4	-1.4	-0.7	-1.3	-3.0	-3.3	0.1	-2.1	-3.3	-0.7	1.3	-2.1	-3.1	-1.6	-0.3
Y	-2.4	-2.4	-0.4	-1.9	-1.6	1.6	-2.6	-2.0	-2.0	-1.0	-0.6	-2.4	-1.0	0.1	-1.0	-2.1	-1.9	-1.1	-2.0	-0.1
S	0.0	-0.1	-2.0	0.0	0.1	-1.1	0.4	0.0	-0.1	-2.0	-1.6	0.4	0.0	-2.0	1.1	0.0	0.0	-1.6	0.7	0.1
G	-1.1	-0.6	-3.3	-1.7	0.0	-2.3	3.4	-0.4	-2.0	-3.3	-3.1	1.0	-1.3	-3.4	1.0	-1.3	-0.3	-3.0	0.0	-1.7
W	-3.4	-3.1	-2.1	-2.9	-0.7	-2.0	-2.4	-3.0	-2.9	-3.0	-2.0	-2.9	-2.1	-1.6	-2.0	-3.4	-2.9	-2.7	-2.9	-2.4
N	-1.7	0.0	-3.0	-0.3	-1.4	0.4	-0.9	0.1	0.0	-3.0	-2.9	0.7	-2.0	-3.0	0.9	0.4	-2.0	-2.9	-1.1	0.6
M	-1.6	-2.0	1.1	-1.0	-1.3	-1.4	-2.1	-1.3	-1.0	1.0	1.0	-2.0	-0.3	1.0	-1.7	-1.6	-0.7	1.0	-1.0	-1.4
A	1.1	-1.1	-0.6	-0.3	1.1	-1.9	1.7	-0.1	-0.9	-0.3	-0.3	-0.1	1.7	-1.0	-0.3	-0.6	2.0	0.1	1.6	-1.1
I	-2.1	-3.1	2.3	-2.7	-2.0	-2.3	-2.7	-2.0	-2.7	3.3	1.7	-2.7	-0.7	2.0	-2.4	-1.9	0.0	2.3	-1.1	-2.9
H	-1.7	-1.0	-2.7	-0.7	-1.9	6.4	-2.0	-1.1	-0.9	-3.0	-2.6	-1.3	-1.7	-2.4	-0.7	-1.0	-2.3	-2.9	-1.9	3.1

→ Summing up those values gives a score of 34.714

Application: PSI-BLAST

- Position-specific iterated BLAST (PSI-BLAST) exploits PSSM to be more sensitive to distantly-related sequences.

PSI-BLAST procedure



Protein family
(Evolutionarily related proteins)

initial top K BLAST hit

Query protein sequence

Compute PSSM with initial top K BLAST hit
& Run PSSM-based search (scoring)

Additional hits from
PSSM-based search

Repeat computing PSSM with top K hits
& PSSM-based search

(Typically, 1-2 iterations provide sufficient sensitivity)

BLAST: Basic Local Alignment Search Tool

HMM as a profile

- As you may have learned in the last lecture, HMM is one of the most powerful models for sequence modeling.

- We can even extend HMM to model shared features of a set of sequences.
  - In other words, we can build an HMM that best summarizes the given set of sequences.

→ Profile HMM (pHMM)

- Meanwhile, HMM needs (1) a set of hidden states, (2) transition probabilities and (3) emission probabilities and pHMM also needs them.

- First of all, what is the hidden states of pHMM?

Hidden states of pHMM

- Although the main goal of pHMM is not for generating sequences, it is easy to understand pHMM model building procedure when we think from a generative point of view.
  - Just for a few slides, think pHMM as a generator of 'likely' sequences that resembles a family of sequences that we are interested in.
  - Specifically, we want a travel along the states (i.e., a complete iteration of state transition+emission from start → end) of pHMM generates a single sequence!

- Imagine that we are now naively aligning a query sequence against a reference alignment. What are the options for the next character?

Query	K	G	?			
Reference alignment	K	G	H	G	K	-
	K	K	H	G	V	T
	E	T	-	A	N	-
	R	W	H	A	E	R
	Q	A	H	A	G	K
	A	A	L	G	A	K

Hidden states of pHMM

1. Match

- If the next character in the query matches well with the reference alignment, we can simply write the character and go ahead.

Query	K	G	H			
Reference alignment	K	G	H	G	K	-
	K	K	H	G	V	T
	E	T	-	A	N	-
	R	W	H	A	E	R
	Q	A	H	A	G	K
	A	A	L	G	A	K

Hidden states of pHMM

2. Insertion

- If the next character in the query is not likely to be derived from the next column of reference alignment, one of the options is to treat the character as an insertion relative to the ref. alignment.

Query	K	G	W			
Reference alignment	K	G	-	H	G	K
	K	K	-	H	G	V
	E	T	-	-	A	N
	R	W	-	H	A	E
	Q	A	-	H	A	G
	A	A	-	L	G	A

Hidden states of pHMM

3. Deletion

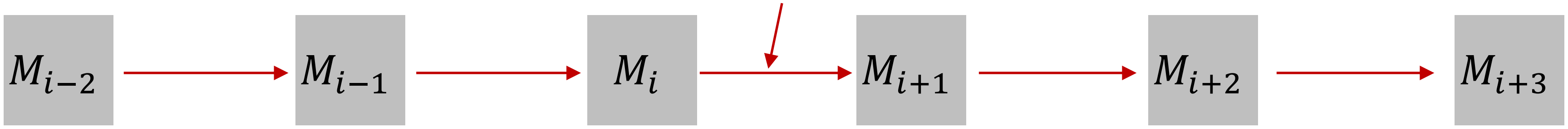
- If the next character in the query is not likely to be derived from the next column of reference alignment, one of the options is to think that the next character had been deleted in the query.

Query	K	G	-	G		
Reference alignment	K	G	H	G	K	-
	K	K	H	G	V	T
	E	T	-	A	N	-
	R	W	H	A	E	R
	Q	A	H	A	G	K
	A	A	L	G	A	K

Hidden states of pHMM

1. Match state
2. Insertion state
3. Deletion state

- Features
  - ✓ Each match state M_i emits a character

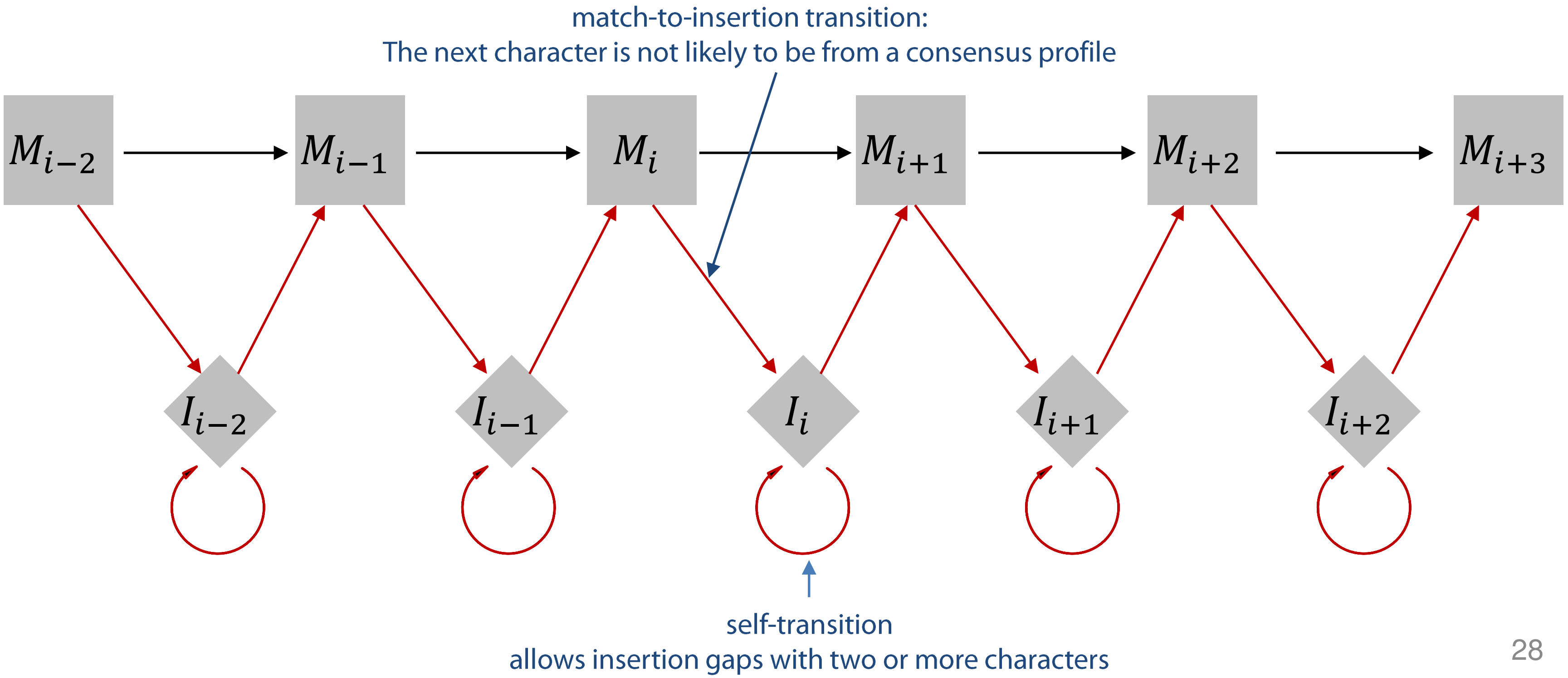


match-to-match transition: the next character matches well with the consensus profile
M_{i-2}
M_{i-1}
M_i
M_{i+1}
M_{i+2}
M_{i+3}

Hidden states of pHMM

1. Match state
2. Insertion state
3. Deletion state

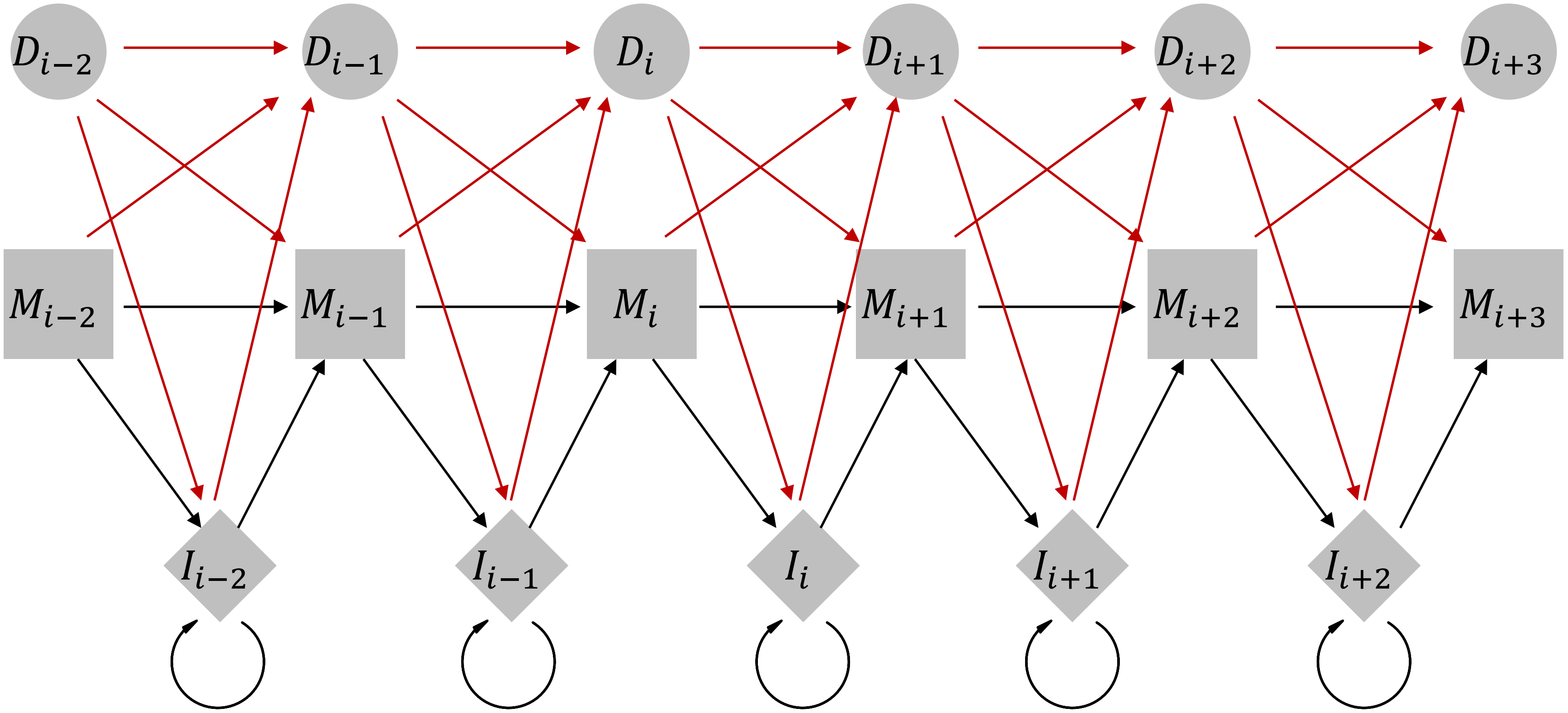
- Features
  - ✓ Each insertion state I_i emits a character
  - ✓ Self-transition exists
  - ✓ But no transition between I_i and I_{i+1} exists


match-to-insertion transition:
The next character is not likely to be from a consensus profile
M_{i-2}
M_{i-1}
M_i
M_{i+1}
M_{i+2}
M_{i+3}
I_{i-2}
I_{i-1}
I_i
I_{i+1}
I_{i+2}
self-transition
allows insertion gaps with two or more characters

Hidden states of pHMM

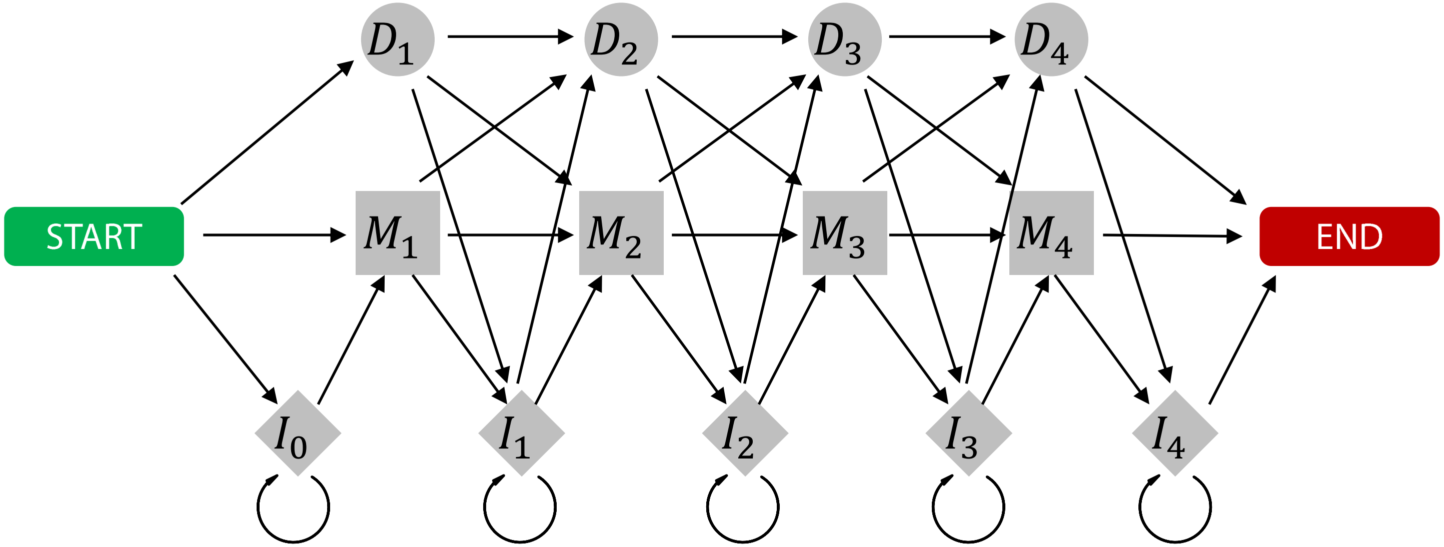
1. Match state
2. Insertion state
3. Deletion state

- Features
  - ✓ Each deletion state D_i does not emit any character
    - ✓ We call it 'silent state'



Q. pHMM can be still be implemented without deletion state. How can it be possible?
If so, what is the benefit of considering separate deletion states?

A complete pHMM structure



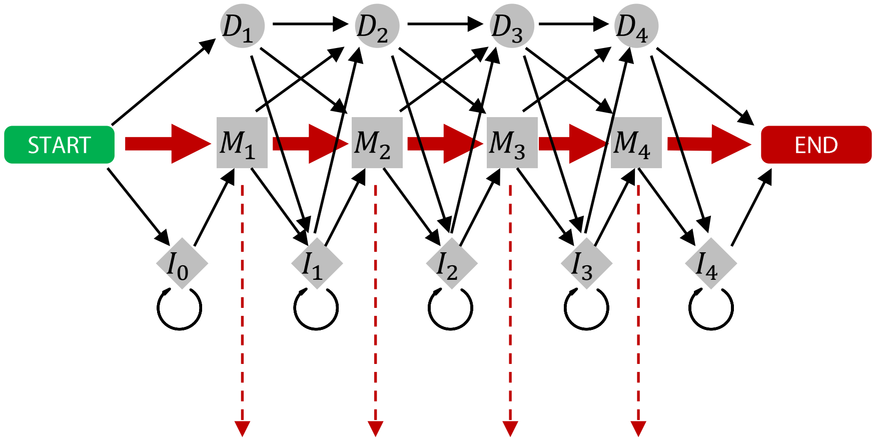
A travel along pHMM produces a sequence

Example 1

Sequences used to build pHMM

K	G	H	G
K	K	H	G
E	T	-	A
R	W	H	A
Q	A	H	A
A	A	L	G

Build pHMM (Next slide)

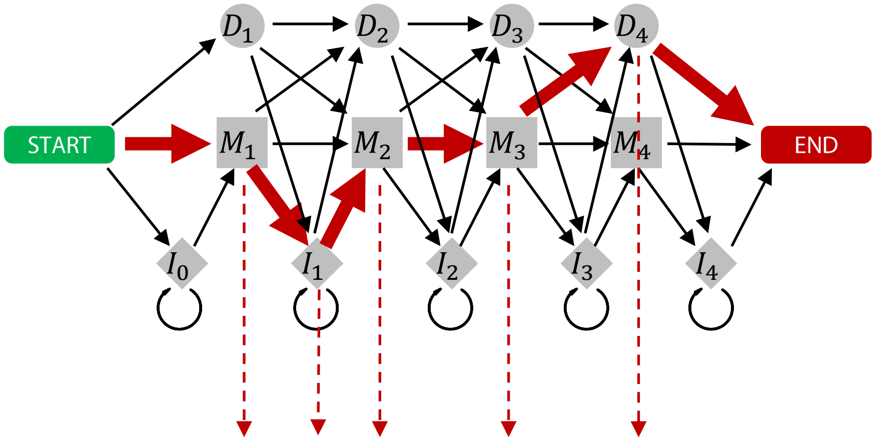


Seq emitted K A H G

Example 2

K	G	H	G
K	K	H	G
E	T	-	A
R	W	H	A
Q	A	H	A
A	A	L	G

Build pHMM (Next slide)



K T A H -

Three questions in pHMM — revisited

- Evaluation
  - What is a probability of generating an observed sequence x by pHMM?
  - Forward / Backward algorithm

- Decoding
  - Given an observed sequence x, what is the most probable hidden state path of pHMM?
  - Viterbi algorithm

- Learning
  - Given a set of sequence S, estimate model parameters of pHMM.
  - Two scenarios:
    - Learning from aligned sequences : Maximum likelihood estimation
    - Learning from unaligned sequences : Baum-Welch algorithm

Three questions in pHMM — revisited

- Evaluation
  - What is a probability of generating an observed sequence x by pHMM?
  - Forward / Backward algorithm

2
Secondly, we will discuss how pHMM can be used for evaluation

- Decoding
  - Given an observed sequence x, what is the most probable hidden state path of pHMM?
  - Viterbi algorithm
- Learning
  - Given a set of sequence S, estimate model parameters of pHMM.
  - Two scenarios:
    - Learning from aligned sequences : Maximum likelihood estimation
    - Learning from unaligned sequences : Baum-Welch algorithm

We will discuss this first!
1

3
Finally, we will briefly discuss how the pHMM parameters can be estimated from unaligned sequences

(Learning) pHMM parameter estimation from multiple sequence alignments

- The parameters of pHMM includes:
  - Transition probability a_{kl} : probability of the transition from state k to l
  - Emission probability $e_k(a)$: probability of emitting character a from state k

- How can we estimate the values of those parameters?

- Recall how we estimated the parameter of HMM.
  - If we know the hidden state path of the sequence, it is straightforward: Maximum likelihood estimation of parameters
  - Tip: You can think of maximum likelihood estimation = event counting

- Now, let's come back to pHMM. How can we know the hidden state path of the sequence?
  - We can derive it from the multiple sequence alignment!

Determining hidden states from multiple sequence alignment

- Labelling each column of multiple sequence alignment as one of match (M) or insertion (I) is often done heuristically.
  - No more than a half gap characters in 'match' state.
  - …

Obvious match state

Obvious insertion state

	A	Q	V	K	G	H	G	K	K	V	A	D	A	L	T	N	A	V	A	H
	P	-	V	K	-	H	G	K	K	V	L	G	A	-	-	D	G	L	A	H
	E	-	L	K	-	H	G	V	T	V	L	T	A	L	G	A	-	L	K	K
	A	P	F	E	-	H	A	N	R	-	V	-	F	-	-	K	I	I	G	E
	A	-	V	R	-	H	A	E	R	I	I	-	-	-	N	D	A	V	A	-
	P	-	L	Q	-	H	A	G	K	V	F	-	L	V	Y	E	A	A	I	Q
	P	-	V	A	-	L	G	A	K	V	L	A	Q	-	G	V	A	V	S	H
States	M	I	M	M	I	M	M	M	M	M	M	M	M	I	M	M	M	M	M	M

Determining hidden states from multiple sequence alignment (cont'd)

- Once column-level hidden states are determined, we can determine the hidden state path for each sequence in the alignment.
  - Match (M), Insertion (I), Deletion (D)

Column-level States	M	I	M	M	I	M	M	M	M	M	M	M	M	I	M	M	M	M	M	M
Seq 1	A	Q	V	K	G	H	G	K	K	V	A	D	A	L	T	N	A	V	A	H
State path for Seq 1	M	I	M	M	I	M	M	M	M	M	M	M	M	I	M	M	M	M	M	M
Seq 2	P	-	V	K	-	H	G	K	K	V	L	G	A	-	-	D	G	L	A	H
State path for Seq 2	M	-	M	M	-	M	M	M	M	M	M	M	M	-	D	M	M	M	M	M
Seq 3	E	-	L	K	-	H	G	V	T	V	L	T	A	L	G	A	-	L	K	K
State path for Seq 3	M	-	M	M	-	M	M	M	M	M	M	M	M	I	M	M	D	M	M	M

And so on…

pHMM parameter estimation from multiple sequence alignments

- Given hidden state paths, it is straightforward to estimate transition and emission probabilities.
  - Again, counting!
  - Note that you may need pseudocounts to overcome the lack of the data.

transitions from state k to state l

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}}$$

total transitions starting from state k

emissions of character a from state k

$$e_k(a) = \frac{E_k(a)}{\sum_{a'} E_k(a')}$$

total emissions from state k

(Decoding) Viterbi decoding

- What is the most probable hidden state path of observed sequence x in pHMM?
  - This is equivalent to "What is the best-scoring alignment between x and the profile?". Why?

- Since we have three different types of states, Viterbi DP recurrence relation needs three separate tables for each state type: $V_j^M(i), V_j^I(i), V_j^D(i)$

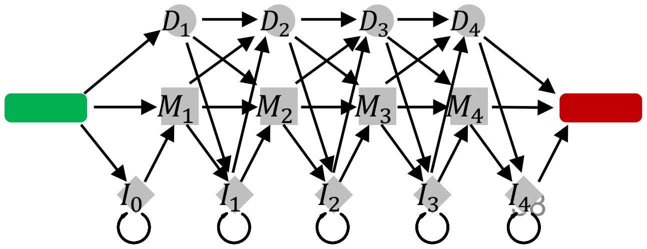
- Given the tables, recurrence relations are:

$$V_j^M(i) = \log \frac{e_{M_j}(x_i)}{q_{x_i}} + max \begin{cases} V_{j-1}^M(i-1) + \log a_{M_{j-1}M_j} \\ V_{j-1}^I(i-1) + \log a_{I_{j-1}M_j} \\ V_{j-1}^D(i-1) + log a_{D_{j-1}M_j} \end{cases}$$

$$V_j^I(i) = \log \frac{e_{I_j}(x_i)}{q_{x_i}} + max \begin{cases} V_j^M(i-1) + \log a_{M_jI_j} \\ V_j^I(i-1) + \log a_{I_jI_j} \\ V_j^D(i-1) + log a_{D_jI_j} \end{cases}$$

We can assume this term as 0, why?

$$V_j^D(i) = max \begin{cases} V_{j-1}^M(i) + \log a_{M_{j-1}D_j} \\ V_{j-1}^I(i) + \log a_{I_{j-1}D_j} \\ V_{j-1}^D(i) + log a_{D_{j-1}D_j} \end{cases}$$



(Evaluation) Forward algorithm

- What is a probability of generating an observed sequence x by pHMM?

- Similarly, we consider three separate tables for memoization: $F_j^M(i)$, $F_j^I(i)$, $F_j^D(i)$

- Given the tables, recurrence relations are:

$$
\begin{aligned}
&F_j^M(i) \\
&= \log \frac{e_{M_j}(x_i)}{q_{x_i}} \\
&+ \log[a_{M_{j-1}M_j} \exp(F_{j-1}^M(i-1)) + a_{I_{j-1}M_j} \exp\left(F_{j-1}^I(i-1)\right) + a_{D_{j-1}M_j} \exp(F_{j-1}^D(i-1))]
\end{aligned}
$$

$$
\begin{aligned}
&F_j^I(i) \\
&= \log \frac{e_{I_j}(x_i)}{q_{x_i}} + \log[a_{M_jI_j} \exp\left(F_j^M(i-1)\right) + a_{I_jI_j} \exp\left(F_j^I(i-1)\right) + a_{D_jI_j} \exp\left(F_j^D(i-1)\right)]
\end{aligned}
$$

$$F_j^D(i) = \log[a_{M_{j-1}D_j} \exp\left(F_{j-1}^M(i)\right) + a_{I_{j-1}D_j} \exp\left(F_{j-1}^I(i)\right) + a_{D_{j-1}D_j} \exp(F_{j-1}^D(i))]$$

(Learning) pHMM parameter estimation from unaligned sequences

- When the given set of sequences are not aligned, we cannot know hidden state paths.

- Use Baum-Welch Expectation-Maximization learning!

- E-step: Compute expected counts of transitions and emissions.
- M-step: Maximum likelihood estimation of a_{kl} and $e_k(a)$

Any questions?