Advanced Database Searching: Sequence Patterns, Profiles & Hidden Markov Models

BI 527, Lecture #14, Fall 2011

Barry Grant
2055A Palmer Commons
Tel: 647-3113
bjgrant@umich.edu
http://thegrantlab.org
Recap of the last lecture: #13

- **Sequence motifs and patterns**
  Finding functional cues from sequence conservation
  Patterns that describe a motif using a qualitative regular expression sequence
    - \[[LFI]\]-x-G-[PT]-P-G-x-G-K-[TS]-[AGSI]\]
  Defining and using patterns and their limitations

- **Sequence profiles**
  Profiles describe a motif using quantitative information captured in a PSSM
  Building log-likelihood ratio PSSMs
  The average score method for protein PSSMs
  Scoring sequences and searching with profiles

- **PSI-BLAST** algorithm
  Iterative PSSM searching to improve BLAST search sensitivity
  PSSM advantages and limitations
  The danger of PSSM corruption (triangular inequality)

- Profile software and databases
Outline of this lecture: #14

- **Major PSSM limitations**
  - Do not capture positional dependencies
  - Hard to recognize pattern instances that contain indels
  - Do not handle boundary detection problems well

- Modeling motifs using **Markov chains**:  
  Pros and cons of Markov models

- **Hidden Markov models (HMMs)**
  - More versatile full probabilistic model for detection of remote similarities
  - Architecture and parameterization
  - Boundary detection
  - Key algorithms: Viterbi, Forward and Baum-Welch algorithms
  - Scoring sequences and generating MSAs
  - HMM limitations

- **HMM software and databases**
  - Summary and example usage
Position Specific Scoring Matrices

A sequence profile is a **position-specific scoring matrix** (or **PSSM**) that gives a **quantitative** description of a sequence motif.

A simple PSSM is a Log odds scoring matrix that has as many columns as there are positions in the alignment, and either 4 rows (one for each DNA nucleotide) or 20 rows (one for each amino acid).

\[
M_{kj} = \log \left( \frac{p_{kj}}{p_j} \right)
\]

- \(M_{kj}\) score for the \(j\)th nucleotide at position \(k\)
- \(p_{kj}\) probability of nucleotide \(j\) at position \(k\)
- \(p_j\) “background” probability of nucleotide \(j\)

See Lecture #13
Problems with PSSMs

PSSMs work well for fixed length motifs in which the sites are more or less independent - i.e., ungapped motifs.

However there are other kinds of motifs for which PSSMs are not well suited. PSSMs cannot:

1. model positional dependencies
2. recognize pattern instances containing insertions or deletions
3. model variable length patterns
4. detect boundaries

\[ \sum_{l=0}^{w-1} M[t[i + l], l] \]
Problems with PSSMs: 1. Positional dependencies

Do not capture positional dependencies

Note: We never see QD or RH, we only see RD and QH. However, P(RH)=0.24, P(QD)=0.24, while P(QH)=0.16
Problems with PSSMs: 2. Insertions and deletions

Hard to recognize pattern instances that contain indels

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>2.4</td>
</tr>
<tr>
<td>E</td>
<td>0.6</td>
<td>2.9</td>
<td>0.6</td>
<td>0.6</td>
<td>1.6</td>
</tr>
<tr>
<td>H</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>3.0</td>
</tr>
<tr>
<td>I</td>
<td>0.8</td>
<td>0.8</td>
<td>3.1</td>
<td>0.8</td>
<td>0.8</td>
</tr>
<tr>
<td>Q</td>
<td>1.1</td>
<td>1.1</td>
<td>1.1</td>
<td>2.1</td>
<td>1.1</td>
</tr>
<tr>
<td>R</td>
<td>0.8</td>
<td>0.8</td>
<td>0.8</td>
<td>2.8</td>
<td>0.8</td>
</tr>
<tr>
<td>W</td>
<td>5.0</td>
<td>2.7</td>
<td>2.7</td>
<td>2.7</td>
<td>1.8</td>
</tr>
</tbody>
</table>

\[
\text{WETIRD} \\
5.0 + 2.9 + 1.2 + 1.4 + 1.5 = 11
\]

\[
\text{WETIRD} \\
1.2 + 1.8 + 3.1 + 3.0 + 3.4 = 12.5
\]

\[
\text{WETIRD} \\
5.0 + 2.9 + 3.1 + 3.0 + 3.4 = 18.4
\]
Problems with PSSMs: 3. Variable length motifs

Cannot easily deal with variable length motifs

WETIRD
WE-IRD
WEIRD
WETIQH
WE-IRD
WETIQH

Gaps can be represented by expanding $\Sigma$ but what size window should be used to score new instances of the motif???
Problems with PSSMs: 4. Detecting boundaries

Do not handle boundary detection problems well
E.g. Label every element in the sequence with a 0 (not in pattern) or a 1 (in pattern)

Examples of boundary detection problems include:
- Recognition of regulatory motifs
- Recognition of protein domains
- Intron/exon boundaries
- Gene boundaries
- Transmembrane regions
- Secondary structure elements (helices and strands)
These shortcomings of PSSMs set the stage for a new kind of profile, based on **Markov chains**, called **Hidden Markov models (HMMs)**

- modeling positional dependencies
- recognizing pattern instances with indels
- modeling variable length patterns
- detecting boundaries
Markov chains

**Markov chains** are stochastic processes that undergo **transitions** between a finite series of **states** in a chainlike manner.

The system transverses states with probability

\[ p(x_1, x_2, x_3, ...) = p(x_1) p(x_2 | x_1) p(x_3 | x_2) p(x_4 | x_3) ... \]

i.e. **Markov chains are memoryless**: the probability that the chain is in state \( x_i \) at time \( t \), depends only on the state at the previous time step and not on the past history of the states visited before time \( t-1 \).

This specific kind of "memorylessness" is called the **Markov property**.

The **Markov property** states that the conditional probability distribution for the system at the next step (and in fact at all future steps) depends only on the current state of the system, and not additionally on the state of the system at previous steps.
Markov chains...

Markov chains, and their extension hidden Markov models (HMMs), are commonly represented by **state diagrams**, which consist of **states** and connecting **transitions**.

![State Diagram Example](image)

E.g., A general Markov chain modeling DNA. Note that any sequence can be traced through the model by passing from one state to the next via the transitions.

A **transition probability** parameter \(a_{ij}\) is associated with each transition (arrow) and determines the probability of a certain state \(S_j\) following another state \(S_i\).

A Markov chain is defined by:
- a finite set of **states**, \(S_1, S_2 \ldots S_N\)
- a set of **transition probabilities**: \(a_{ij} = P(q_{t+1}=S_j|q_t=S_i)\)
- and an **initial state probability distribution**, \(\pi_i = P(q_0=S_i)\)
Simple Markov chain example for \( x = \{a, b\} \)

Observed sequence: \( x = \text{abaaababbaa} \)

Model:

\[
P(x) = 0.5 \times 0.3 \times 0.5 \times 0.7 \times 0.7 \times 0.3 \times 0.5 \times 0.3 \times 0.5 \times 0.5 \times 0.7
\]

Q. Can you sketch the state diagram with labeled transitions for this model?
Typical questions we can ask with Markov chains include:
• What is the probability of being in a particular state at a particular time? 
  (By time here we can read position in our query sequence)

• What is the probability of seeing a particular sequence of states? 
  (I.e., the score for a particular query sequence given the model)

Q. What do Markov chains add over the traditional PSSM approach? 
In particular how do Markov chains deal with the following PSSM weaknesses?

1. Positional dependencies
2. Pattern instances containing insertions or deletions
3. Variable length patterns, and
4. The detection boundaries (i.e. segmentation of sequences)
Markov chains: 1. Positional dependencies

The connectivity or **topology** of a Markov chain can easily be designed to capture dependencies and variable length motifs.

Recall that a PSSM for this motif would give the sequences **WEIRD** and **WEIRH** equally good scores even though the RH and QR combinations were not observed.
To address pattern instances with gaps and variable length motifs, we can construct a Markov chain to recognize a query sequence with insertions (via an extra insertion state) and deletions (via extra transitions (edges)).
Markov chains: 3. Boundary detection

Giving a sequence we wish to label each symbol in the sequence according to its class (e.g. transmembrane regions or extracellular/cytosolic).

Given a training set of labeled sequences we can begin by modeling each amino acid as hydrophobic (H) or hydrophilic (L) i.e. reduce the dimensionality of the 20 amino acids into two classes.

E.g., A peptide sequence can be represented as a sequence of Hs and Ls. e.g. HHHLLLHHLHHLHLHLH...
Markov chains: boundary detection...

A simpler question: **is a given sequence a transmembrane sequence?**

A Markov chain for recognizing transmembrane sequences

- States: $S_H$, $S_L$
- $\Sigma = \{H, L\}$
- $\pi(H) = 0.6$, $\pi(L) = 0.4$

Question: Is sequence **HHLHH** a transmembrane protein?

$$P(HHLHH) = 0.6 \times 0.7 \times 0.7 \times 0.3 \times 0.7 \times 0.7 = 0.043$$

Problem: need a threshold,
threshold must be length dependent
Markov chains: boundary detection

We can classify an observed sequence \((O = O_1, O_2, \ldots)\) by its log odds ratio.

### Transmembrane Model

- \(\pi(H) = 0.6\), \(\pi(L) = 0.4\)

\[
\begin{align*}
P(HHLHH \mid TM) &= 0.6 \times 0.7 \times 0.7 \times 0.3 \times 0.7 \times 0.7 \\
&= 0.043
\end{align*}
\]

### Null Model

- \(\pi(H) = 0.5\), \(\pi(L) = 0.5\)

\[
\begin{align*}
P(HHLHH \mid EC) &= 0.5 \times 0.5 \times 0.5 \times 0.5 \times 0.5 \times 0.5 \\
&= 0.016
\end{align*}
\]

\[
\frac{P(HHLHH \mid TM)}{P(HHLHH \mid EC)} = \frac{0.043}{0.016} = 2.69
\]

In other words, it is more than twice as likely that \textbf{HHLHH} is a transmembrane sequence. The log-odds score is: \(\log_2(2.69) = 1.43\)
Side note: Parameter estimation

Both initial probabilities ($\pi(i)$) and transition probabilities ($a_{ij}$) are determined from known examples of transmembrane and non-transmembrane sequences.

- initial probabilities $\pi(H), \pi(L)$
- transition probabilities: $a_{HH}, a_{HL}, a_{LH}$ and $a_{LL}$.

Given labeled sequences (TM and E/C), we determine the initial probabilities $\pi(i)$ by counting the number of sequences that begin with residue $i$.

To determine transition probabilities, $a_{ij}$, we first determine $A_{ij}$ (the number of transitions from state $i$ to $j$ in the training data, i.e. count the number of $ij$ pairs in the training data). Then normalize by the number of $i^*$ pairs.

$$a_{ij} = \frac{A_{ij}}{\sum_j A_{ij}}$$
Side note: Parameter estimation...

Both initial probabilities ($\pi(i)$) and transition probabilities ($a_{ij}$) are determined from known examples of transmembrane and non-transmembrane sequences.

\[ a_{HL} = \frac{A_{HL}}{\sum_i A_{Hi}} \]

$\pi(H) = \#$ of sequences that begin with H, normalized by the total $\#$ of training sequences

- $\pi(H) = 0.6$, $\pi(L) = 0.4$

HH... (A_{HL} = 12, A_{H*} = 40)
Boundary detection challenge

**Given sequence of Hs and Ls, find all transmembrane regions:**

Using our Markov models we would still need to score successive overlapping windows along the sequence, leading to a fuzzy boundary (just as with a PSSM).

To approach this question we can construct a new four state model by adding transitions connecting the TM and E/C models.

Transitions between the $M$ states and the $E/C$ states indicate boundaries between membrane regions and cytosolic or extracellular regions.

However this is no longer a standard Markov chain!
In a Markov chain, there is a one-to-one correspondence between symbols and states, which is not true of our new merged four state, two symbol model.

For example, both $H_M$ and $H_{E/C}$ are associated with hydrophilic residues.

- This four-state transmembrane model is a **hidden Markov model**.

\[
\begin{array}{c}
\begin{array}{c}
\text{HM} \\
0.6 \\
0.1 \\
0.4
\end{array} \\
\begin{array}{c}
\text{L}_M \\
0.6 \\
0.2 \\
0.4
\end{array} \\
\begin{array}{c}
\text{HE/C} \\
0.1 \\
0.1 \\
0.4
\end{array} \\
\begin{array}{c}
\text{LE/C} \\
0.1 \\
0.1 \\
0.4
\end{array}
\end{array}
\]
So what's hidden?

We will distinguish between the observed parts of the problem and the hidden parts.

- In the Markov models we have considered previously it is clear which states account for each part of the observed sequence.
  Due to the one-to-one correspondence between symbols and states.

- In our new model, there are multiple states that could account for each part of the observed sequence.
  i.e. we don’t know which state emitted a given symbol from knowledge of the sequence and the structure of the model.
  - This is the hidden part of the problem.
For our Markov models
• Given HLLH..., we know the exact state sequence (q₀=S_H, q₁=S_L, q₂=S_L, ...)

For our HMM
• Given HLLH..., we must infer the most probable state sequence
• This HMM state sequence will yield the boundaries between likely TM and E/C regions
Side note: HMM states as sequence emitters

It’s useful to imagine HMM states **emitting symbols** each time they are visited.

In this way, transversing the model will “generate” a sequence with a certain probability (i.e. “score”).

This probability is a product of the state path taken through the model. That is, it depends on **initial probabilities**, **transition probabilities** and **emission probabilities** (the probability that a visited state emits a particular symbol) along the path.

There may be many possible paths that can generate the same sequence.

An HMM is a **full probabilistic model** – the model parameters $\theta$ and the overall sequence “scores” $P(x, S \mid HMM, \theta)$ are all probabilities. As a result, we can use standard **Bayesian probability theory** to manipulate these numbers in powerful ways, including optimizing parameters, calculating confidence in predictions, and interpreting the statistical significance of scores.
Hidden Markov models (HMMs)

Markov Chains
- States: $S_1, S_2, \ldots, S_N$
- Initial probabilities: $\pi_i$
- Transition probabilities: $a_{ij}$

Hidden Markov Models
- States: $S_1, S_2, \ldots, S_N$
- Initial probabilities: $\pi_i$
- Transition probabilities: $a_{ij}$
- Alphabet of emitted symbols, $\Sigma$
- Emission probabilities: $e_i(a)$
  probability state $i$ emits symbol $a$

One-to-one correspondence between states and symbols

Symbol may be emitted by more than one state

Similarly, a state can emit more than one symbol
Example three state HMM

In this example we will use only one state for the transmembrane segment (M) and use emission probabilities to distinguish between H and L residues. We will also add separate E & C states with distinct emission probabilities.

\[ a_{ij} = \begin{bmatrix}
0.7 & 0.3 & 0 \\
0.25 & 0.5 & 0.25 \\
0 & 0.3 & 0.7
\end{bmatrix} \]
Side note: Parameter estimation

As in the case of Markov chains, the HMM parameters can be learned from labeled training data.

Note that we now have to learn the initial probabilities, transition probabilities and emission probabilities.

\[
a_{ij} = \frac{A_{ij}}{\sum_{j'} A_{ij'}}
\]

\[
e_i(x) = \frac{E_i(x)}{\sum_{x'} E_i(x')}
\]

\[
\begin{array}{c|c|c|}
\pi_i & E & M & C \\
\hline
\pi_0 & 0 & 0 & 1 \\
\hline
e_i(H) & 0.2 & 0.9 & 0.3 \\
e_i(L) & 0.8 & 0.1 & 0.7 \\
\end{array}
\]
### Query Sequence

<table>
<thead>
<tr>
<th>States</th>
<th>H</th>
<th>H</th>
<th>L</th>
<th>L</th>
<th>L</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>E</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>M</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>C</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>START</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\[ \pi(E) = 0 \]
\[ \pi(M) = 0 \]
\[ \pi(C) = 1 \]

\[
\begin{array}{c|c|c|c}
\text{States} & \text{H} & \text{H} & \text{L} & \text{L} & \text{H} \\
\hline
E & 0 \times 0.2 & =0 & & & \\
M & 0 \times 0.9 & =0 & & & \\
C & 1 \times 0.3 & =0.3 & & & \\
\end{array}
\]

Query Sequence

- E: H 0.2, L 0.8
- M: H 0.9, L 0.1
- C: H 0.3, L 0.7
<table>
<thead>
<tr>
<th>States</th>
<th>H</th>
<th>H</th>
<th>L</th>
<th>L</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>(0 \times 0.2 = 0)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>(0 \times 0.9 = 0)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>(1 \times 0.3 = 0.3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[\pi(E) = 0\]
\[\pi(M) = 0\]
\[\pi(C) = 1\]
The image contains a graphical representation of a Markov model with states E, M, and C. The transitions and probabilities between the states are indicated by arrows and numbers. The states are associated with emission probabilities represented by $e_i$ for each state.

Below the diagram, there is a table labeled "Query Sequence" with columns for the states E, M, and C, and rows for the emission probabilities $H$ and $L$. The table shows the calculation of the emission probabilities for each state.

For state E:
- Emission probability for H is $0 \times 0.2 = 0$.

For state M:
- Emission probability for H is $0 \times 0.9 = 0$.
- Emission probability for L is $0.3 \times 0.9 \times 0.3 = 0.081$.

For state C:
- Emission probability for H is $1 \times 0.3 = 0.3$.
- Emission probability for L is $0.7 \times 0.3 \times 0.3 = 0.063$.

The table also indicates the start state as "START".
### States and Query Sequence

<table>
<thead>
<tr>
<th>States</th>
<th>H</th>
<th>H</th>
<th>L</th>
<th>L</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>0x0.2=0</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>0x0.9=0</td>
<td>0.3x0.9x0.3</td>
<td>0.25</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>1x0.3=0.3</td>
<td>0.7x0.3x0.3</td>
<td>0.25</td>
<td>0.3</td>
<td></td>
</tr>
</tbody>
</table>

- **States**: E, M, C
- **Query Sequence**: H, H, L, L, H
- **Initial State**: START
\[
\begin{align*}
\text{E} &: 0 \times 0.2 = 0, \\
\text{M} &: 0 \times 0.9 = 0, \\
\text{C} &: 1 \times 0.3 = 0.3
\end{align*}
\]
<table>
<thead>
<tr>
<th>States</th>
<th>H</th>
<th>H</th>
<th>L</th>
<th>L</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>0x0.2 =0</td>
<td>-</td>
<td>0.25x0.8x0.081 =0.016</td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>0x0.9 =0</td>
<td>0.3x0.9x0.3 =0.081</td>
<td>0.5x0.1x0.081 =0.04</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>1x0.3 =0.3</td>
<td>0.7x0.3x0.3 =0.063</td>
<td>0.25x0.7x0.081 =0.014</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Query Sequence:

- E: H 0.2, L 0.8
- M: H 0.9, L 0.1
- C: H 0.3, L 0.7

Diagram:

- E: H 0.2, L 0.8
- M: H 0.9, L 0.1
- C: H 0.3, L 0.7

Transition probabilities:

- E to M: 0.25
- M to C: 0.3
- C to E: 0.25
- E to E: 0.7
- M to M: 0.7
- C to C: 0.7
H 0.2
L 0.8

M 0.3
L 0.7

C 0.7
H 0.3
L 0.1

Query Sequence

<table>
<thead>
<tr>
<th>States</th>
<th>H</th>
<th>H</th>
<th>L</th>
<th>L</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>0\times0.2 =0</td>
<td>-</td>
<td>0.25\times0.8\times0.081 =0.016</td>
<td>0.7\times0.8\times0.016 =0.009</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>0\times0.9 =0</td>
<td>0.3\times0.9\times0.3 =0.081</td>
<td>0.5\times0.1\times0.081 =0.04</td>
<td>0.3\times0.1\times0.016 =0.0005</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>1\times0.3 =0.3</td>
<td>0.7\times0.3\times0.3 =0.063</td>
<td>0.25\times0.7\times0.081 =0.014</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

START
%H = 0.2
%H = 0.9
%H = 0.3
\text{START}

\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\text{States} & \text{H} & \text{H} & \text{L} & \text{L} & \text{H} \\
\hline
\text{E} & 0 \times 0.2 & - & 0.25 \times 0.8 \times 0.081 & 0.7 \times 0.8 \times 0.016 & 0.7 \times 0.2 \times 0.009 \\
& = 0 & & = 0.016 & = 0.009 & = 0.001 \\
\hline
\text{M} & 0 \times 0.9 & 0.3 \times 0.9 \times 0.3 & 0.5 \times 0.1 \times 0.081 & 0.3 \times 0.1 \times 0.016 & 0.3 \times 0.9 \times 0.009 \\
& = 0 & = 0.081 & = 0.04 & = 0.0005 & = 0.002 \\
\hline
\text{C} & 1 \times 0.3 & 0.7 \times 0.3 \times 0.3 & 0.25 \times 0.7 \times 0.081 & - & - \\
& = 0.3 & = 0.063 & = 0.014 & & \\
\hline
\end{tabular}
\[
\begin{array}{cccc}
\text{States} & \text{H} & \text{H} & \text{L} & \text{L} & \text{H} \\
\hline
E & 0 \times 0.2 = 0 & - & 0.25 \times 0.8 \times 0.081 = 0.016 & 0.7 \times 0.8 \times 0.016 = 0.009 & 0.7 \times 0.2 \times 0.009 = 0.001 \\
M & 0 \times 0.9 = 0 & 0.3 \times 0.9 \times 0.3 = 0.081 & 0.5 \times 0.1 \times 0.081 = 0.04 & 0.3 \times 0.1 \times 0.016 = 0.0005 & 0.3 \times 0.9 \times 0.009 = 0.002 \\
C & 1 \times 0.3 = 0.3 & 0.7 \times 0.3 \times 0.3 = 0.063 & 0.25 \times 0.7 \times 0.081 = 0.014 & - & - \\
\hline
\text{START} & \text{C} & \text{E} & \text{M} & \text{C} & \text{START}
\end{array}
\]
\[
\begin{align*}
E & \quad 0.25 \quad 0.3 \\
M & \quad 0.5 \quad 0.25 \\
C & \quad 0.7 \quad 0.7
\end{align*}
\]

**States**

<table>
<thead>
<tr>
<th>States</th>
<th>H</th>
<th>H</th>
<th>L</th>
<th>L</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>0x0.2 =0</td>
<td>-</td>
<td>0.25x0.8x0.081 =0.016</td>
<td>0.7x0.8x0.016 =0.009</td>
<td>0.7x0.2x0.009 =0.001</td>
</tr>
<tr>
<td>M</td>
<td>0x0.9 =0</td>
<td>0.3x0.9x0.3 =0.081</td>
<td>0.5x0.1x0.081 =0.04</td>
<td>0.3x0.1x0.016 =0.0005</td>
<td>0.3x0.9x0.009 =0.002</td>
</tr>
<tr>
<td>C</td>
<td>1x0.3 =0.3</td>
<td>0.7x0.3x0.3 =0.063</td>
<td>0.25x0.7x0.081 =0.014</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

**Query Sequence**

- START: C, M, E
### Query Sequence Table

<table>
<thead>
<tr>
<th>States</th>
<th>H</th>
<th>H</th>
<th>L</th>
<th>L</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>$0 \times 0.2$ = 0</td>
<td>-</td>
<td>$0.25 \times 0.8 \times 0.081$ = 0.016</td>
<td>$0.7 \times 0.8 \times 0.016$ = 0.009</td>
<td>$0.7 \times 0.2 \times 0.009$ = 0.001</td>
</tr>
<tr>
<td>M</td>
<td>$0 \times 0.9$ = 0</td>
<td>$0.3 \times 0.9 \times 0.3$ = 0.081</td>
<td>$0.5 \times 0.1 \times 0.081$ = 0.04</td>
<td>$0.3 \times 0.1 \times 0.016$ = 0.0005</td>
<td>$0.3 \times 0.9 \times 0.009$ = 0.002</td>
</tr>
<tr>
<td>C</td>
<td>$1 \times 0.3$ = 0.3</td>
<td>$0.7 \times 0.3 \times 0.3$ = 0.063</td>
<td>$0.25 \times 0.7 \times 0.081$ = 0.014</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>START</td>
<td>C</td>
<td>M</td>
<td>E</td>
<td>E</td>
<td>-</td>
</tr>
</tbody>
</table>

### Diagram

- States: E, M, C
- Edges: E to E (0.7), E to M (0.3), M to C (0.5), C to E (0.3), M to M (0.25), C to C (0.7)
- Transition Probabilities:
  - E to E: 0.7
  - E to M: 0.3
  - M to C: 0.5
  - C to E: 0.3
  - M to M: 0.25
  - C to C: 0.7

### Calculations

- $e_i$: E to H (0.2), E to L (0.8)
- M to H (0.9), M to L (0.1)
- C to H (0.3), C to L (0.7)
### States Transition Diagram

![Diagram showing transitions between states E, M, and C with probabilities 0.25, 0.3, and 0.7, respectively.](image)

### Query Sequence

<table>
<thead>
<tr>
<th>States</th>
<th>H</th>
<th>H</th>
<th>L</th>
<th>L</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>0x0.2 =0</td>
<td>-</td>
<td>0.25x0.8x0.081 =0.016</td>
<td>0.7x0.8x0.016 =0.009</td>
<td>0.7x0.2x0.009 =0.001</td>
</tr>
<tr>
<td>M</td>
<td>0x0.9 =0</td>
<td>0.3x0.9x0.3 =0.081</td>
<td>0.5x0.1x0.081 =0.04</td>
<td>0.3x0.1x0.016 =0.0005</td>
<td>0.3x0.9x0.009 =0.002</td>
</tr>
<tr>
<td>C</td>
<td>1x0.3 =0.3</td>
<td>0.7x0.3x0.3 =0.063</td>
<td>0.25x0.7x0.081 =0.014</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

### Table

<table>
<thead>
<tr>
<th>States</th>
<th>H</th>
<th>H</th>
<th>L</th>
<th>L</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>C</td>
<td>M</td>
<td>E</td>
<td>E</td>
<td>M</td>
</tr>
</tbody>
</table>
States | H | H | L | L | H
--- | --- | --- | --- | --- | ---
E | 0x0.2 =0 | - | 0.25x0.8x0.081 =0.016 | 0.7x0.8x0.016 =0.009 | 0.7x0.2x0.009 =0.001
M | 0x0.9 =0 | 0.3x0.9x0.3 =0.081 | 0.5x0.1x0.081 =0.04 | 0.3x0.1x0.016 =0.0005 | 0.3x0.9x0.009 =0.002
C | 1x0.3 =0.3 | 0.7x0.3x0.3 =0.063 | 0.25x0.7x0.081 =0.014 | - | -
START | C | M | E | E | M

Most Probable State Sequence

Query Sequence

0.25 0.3
0.7 0.7

e_i
H 0.2
L 0.8

e_i
H 0.9
L 0.1

e_i
H 0.3
L 0.7
We have just used the Viterbi algorithm

The **Viterbi algorithm** finds the most probable “state path” \((S^*)\) (i.e. sequence of hidden states) for generating a given sequence \((x = x_1, x_2, \ldots, x_N)\)

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

This process is often called **decoding** because we “decode” the sequence of symbols to determine the hidden sequence of states.

HMMs were originally developed in the field of speech recognition, where speech is “decoded” into words or phonemes to determine the meaning of the utterance.

Note that we could have used brute force by calculating \(P(x|S)\) for all paths but this quickly becomes intractable for longer sequences or HMMs with a large number of states.

The Viterbi algorithm is guaranteed to find the most probable state path given a sequence and an HMM.

See Durbin *et al.* *Biological Sequence Analysis*
Three key HMM algorithms

• **Viterbi algorithm**
  Given observed sequence $x$ and an HMM $M$, composed of states $S$, calculate the most likely state sequence, $S^*$
  
  $S^* = \arg\max P(x,S)$

• **Forward algorithm**
  Given observed sequence $x$ and an HMM composed of states $S$, calculate the probability of the sequence for the HMM, $P(x|M)$
  
  $P(x) = \sum_{S} P(x,S)$

• **Baum-Welch algorithm**
  Given many observed sequences, estimate the parameters of the HMM
  
  heuristic expectation maximization method to optimize of $a_{ij}$ and $e_i(a)$
The forward algorithm

Another important question is how well does a given sequence fit the HMM?

To answer this question we must sum over all possible state paths that are consistent with the sequence in question (Because we don't know which path emitted the sequence)

The number of paths can quickly become intractable. The forward algorithm is a simple dynamic programing solution that makes use of the Markov property so that we don’t have to explicitly enumerate every path.

The forward algorithm basically replaces the maximization step of the Viterbi algorithm with sums to calculate the probability of the sequence given a HMM.

\[ P(x) = \sum_{s} P(x,S) \]

See Durbin et al. Biological Sequence Analysis
The Baum-Welch algorithm

The **Baum-Welch algorithm** is an **heuristic optimization** algorithm for learning probabilistic models in problems that involve hidden states.

If we *know* the state path for each training sequence (i.e. no hidden states with respect to the training sequences), then learning the model parameters is simple (just like it was for Markov chain models):

- count how often each transition and emission occurs
- normalize to get probabilities

If we *don’t know* the path for each training sequence, we can use the **Baum-Welch algorithm**, an expectation maximization method, which estimates counts by considering every path weighted by its probability:

- start from a given initial guess for the parameters
- perform a calculation which is guaranteed to improve the previous guess
- run until there is little change in parameters between iterations

For sequence profile-HMMs we train from a MSA and hence we can *estimate* our probabilities from the observed sequences.
Segmentation/boundary detection

**Given:** A test sequence and a HMM with different sequence classes

**Task:** Segment the sequence into subsequences, predicting the class of each subsequence

**Question:** What is the most probable “path” (sequence of hidden states) for generating a given sequence from the HMM?

**Solution:** Use the Viterbi algorithm

Classification/sequence scoring

**Given:** A test sequence and a set of HMMs representing different sequence classes

**Task:** Determine which HMM/class best explains the sequence

**Question:** How likely is a given sequence given a HMM?

**Solution:** Use the Forward algorithm

Learning/parameterization

**Given:** A model, a set of training sequences

**Task:** Find model parameters that explain the training sequences

**Question:** Can we find a high probability model for sequence characterization

**Solution:** Use the Forward backward algorithm
Segmentation/boundary detection

*Question:* What is the most probable “path” (sequence of hidden states) for generating a given sequence from the HMM?

*HMMER:* **hmmalign** - align sequences to our HMM

Classification/sequence scoring

*Question:* How likely is a given sequence given a HMM?

*HMMER:* **hmmsearch** - find sequences that match our HMM

Learning/parameterisation

*Question:* Can we find a high probability model for sequence characterization

*HMMER:* **hmmbuild** - setup our HMM parameters
Half time break...

Questions:
For what kinds of motifs are PSSMs not well suited?
What is the Markov property?
In what important ways do HMMs differ from Markov chains?
What is the Viterbi algorithm used for?
How does the Forward algorithm differ from the Viterbi algorithm?
**In what important ways do HMMs differ from Markov chains?**

HMMs differ from Markov chains in a number of ways:
- In HMMs, the sequence of states visited is hidden. Unlike Markov Chains, there is no longer a one-to-one correspondence between states and output symbols.
- In a HMM the same symbol may be emitted by more than one state.
- In a HMM a state can emit more than one symbol.

**What is the Markov property?**

The Markov property states that the conditional probability distribution for the system at the next step (and in fact at all future steps) depends only on the current state of the system, and not additionally on the state of the system at previous steps.

**For what kinds of motifs are PSSMs not well suited?**

PSSMs are not well suited to pattern instances containing insertions or deletions, variable length patterns and those with positional dependencies.

**What is the Viterbi algorithm used for?**

The Viterbi algorithm is used to find the most probable state path given a sequence and an HMM.
HMM network structure is hand tailored to the problem

No algorithm for the prediction of optimal HMM network structure and probabilities has yet been able to beat simple hand-built topologies.

These topologies are tailored to the problem at hand - exon/intron detection, transmembrane regions, secondary structure elements, protein families...
GenScan - gene-prediction HMM

Here, each circle or square represents a functional unit (a state) of a gene on its forward strand (for example, $E_{\text{init}}$ is the 5' coding sequence (CDS) and $E_{\text{term}}$ is the 3' CDS, and the arrows represent the transition probability from one state to another. The GenScan HMM is trained by pre-computing the transition probabilities from a set of known gene structures.

TMHMM - transmembrane protein topology prediction

Each box corresponds to one or more states in the HMM. Cyt. represents the cytoplasmic side of the membrane and non-cyt. the other side. (b) The detailed structure of the inside and outside loop models and helix cap models. (c) The structure of the model for the helix core modeling lengths between 5 and 25, which translates to helices between 15 and 35 when the caps are included.

See: Krogh et al. (2001) JMB 305, 567-580
SAMTOOLS - SNP calling in NextGen sequencing data

Application of HMMs in the area of SNP discovery from NextGen sequencing data, to greatly reduce false SNP calls caused by misalignments around insertions and deletions (indels). The central concept is per-Base Alignment Quality, which accurately measures the probability of a read base being wrongly aligned.

See: Li et al. (2011) Bioinformatics 27, 1157–1158
HMMER - protein homology detection and alignment

Profile HMM architecture used in HMMER2, SAM and PFTOOLS protein homology detection and alignment packages. Match states carry position-specific emission probabilities for scoring residues at each consensus position. Insert states emit residues with emission probabilities identical to a background distribution. We will describe this in more detail shortly...

Building sequence profile-HMMs: Match states

How do the above HMMs relate to profiles? Let’s see how we can use the HMM framework to build profile HMMs that describe families of related sequences.

In the last lecture, we built a profile for the alignment:

Ignoring the “background” frequencies for now, a profile for this alignment can be viewed as a simple HMM with one “match” state for each column, where consecutive match states are separated by transitions of probability 1.

**Q. Why is this not a Markov chain?**
Introduce **insert states** \((I_j)\), which will model inserts after the \(j\)th column in our alignment.

Typically, the output probabilities for insert states are set equal to the background probabilities. Note that we can have different probabilities for entering different insert states, and this models the fact that insertions may be less well-tolerated in certain portions of the alignment.
Building profile-HMMs: Insert states + affine gaps

For any particular insert state, we may have different transition probabilities for entering it for the first time vs. staying in the insert state; this models affine gap penalties.
Building profile-HMMs: Delete states

One could model deletions with additional transitions between match states. However, arbitrarily long gaps would introduce lots of transitions in the model. Instead, we will introduce delete states that do not emit any symbols.
Building profile-HMMs

Putting it all together we get a complete profile HMM topology with match, insert and delete states.

However we still need to decide how many states our HMM has, what the transition probabilities are, etc.
Example profile-HMM building

• How do we pick the length of the HMM?
  Common heuristic is to include only those columns that have > 50% occupancy

• How do we pick emission probabilities for match states?
  \[
  b_{m1}(V) = \frac{5}{7} \\
  b_{m1}(F) = \frac{1}{7} \\
  b_{m1}(I) = \frac{1}{7}
  \]

How do we pick transition probabilities?
• We let the transition probability of going from state \(i\) to state \(j\), \(a_{ij}\) be equal to:

  \[
  \frac{\text{No. of transitions from state } i \text{ to state } j}{\text{No. of transitions from state } i \text{ to any other state}}
  \]

  \[
  a_{M2M3}(V) = \frac{6}{7} \quad \text{No. of matches (=6)} \\
  a_{M2D3}(F) = \frac{1}{7} \quad \text{No. of gaps (=1)} \\
  a_{M2I2}(I) = 0/7 \quad \text{No. of insertions (=0)}
  \]
Side note: Weighting the training sequences

If there is a high degree of **redundancy** in our initial MSA (i.e. it contains a large group of very closely related sequences and a small number of more distantly related sequences) the resulting HMM will over represent the similar sequences and adversely effect our ability to detect distantly related sequences when searching databases.

**Sequences weighting** attempts to compensate for this **sequence sampling bias** by differentially weighting sequences to reduce redundancy prior to model building.

By default HMMER uses a sequence clustering tree as a guide to weight each sequence by its distance to other sequences. This approach will effectively down-weight the influence of redundant sequences.

A number of other approaches have been developed (Voronoi algorithm, maximum entropy, etc.)

See: Karchin et al. (1998) Bioinformatics 14, 772-778
Side note: Pseudocounts and Dirichlet distributions

Unfortunately, for alignments containing a small number of sequences the observed counts may not be representative of the family as a whole.

In such cases we must adjust the probabilities to account for our under-sampling (i.e. unobserved residues)

One common approach is to add pseudocounts to the observed counts so that no zero probabilities can occur.

Simplest approach is to just add one to all counts. More accurate adjustments consider prior knowledge about the behavior of sequence families adjusting counts according to pre-tabulated Dirichlet distributions - which are rather like protein comparison matrixes used in profile methods

Such information is often called prior information, indicating that it is known before any sequence data is seen

See: Durbin et al. “Biological Sequence Analysis”
Generating multiple sequence alignments

Large MSAs can be generated very quickly by using the Viterbi algorithm to find the most likely path through the HMM for a set of unaligned sequences.

This is the basis of the PFAM database which uses the HMMER software package. Namely, HAMMER’s `hmmalign` from the results of `hmmsearch`.

MSA produced by HMMs are not true MSAs in the way that those produced by ClustalW are. ClustalW compares every sequence to every other sequence, whereas HMM aligning compares every sequence to the model independently so that the alignment between sequences is by proxy. Adding new sequences to the ClustalW alignment will add new information which may alter the alignment of existing sequences; adding new sequences to the HMM alignment never changes the alignment of any sequences relative to each other.

As an alternative to HMMER, you can use the Sequence Alignment and Modeling Software System (SAM)

http://compbio.soe.ucsc.edu/sam.html
Recent speed benchmarks indicate that HMMER3 is approaching BLAST speed.

Each point represents a speed measurement for one search with one query against target sequences. Both axes are logarithmic, for speed in millions of dynamic programming cells per second (Mc/s) on the y-axis and query length on the x-axis.

HMM sequence searching performance...

However HMMER3 has a much higher search sensitivity and specificity

In each benchmark, true positive subsequences have been selected to be no more than 25% identical to any sequence in the query alignment ... (see paper for details).

HMM limitations

HMMs are linear models and are thus **unable to capture higher order correlations** among positions (e.g. distant cysteins in a disulfide bridge, RNA secondary structure pairs, etc).

Another flaw of HMMs lies at the very heart of the mathematical theory behind these models. Namely, that the probability of a sequence can be found from the product of the probabilities of its individual residues.

This claim is only valid if the probability of a residue is independent of the probabilities of its neighbors. In biology, there are frequently **strong dependencies between these probabilities** (e.g. hydrophobic residues clustering at the core of protein domains).

These biological realities have motivated research into new kinds of statistical models. These include hybrids of HMMs and neural nets, dynamic Bayesian nets, factorial HMMs, Boltzmann trees and stochastic context-free grammars.

See: Durbin et al. “Biological Sequence Analysis”
**PFAM**: Protein Family Database of Profile HMMs

Comprehensive compilation of both multiple sequence alignments and profile HMMs of protein families.

http://pfam.sanger.ac.uk/

PFAM consists of two databases:
- **Pfam-A** is a manually curated collection of protein families in the form of multiple sequence alignments and profile HMMs. HMMER software is used to perform searches.
- **Pfam-B** contains additional protein sequences that are automatically aligned. Pfam-B serves as a useful supplement that makes the database more comprehensive.
- Pfam-A also contains higher-level groupings of related families, known as clans.
Pfam 25.0 (March 2011, 12273 families)

The Pfam database is a large collection of protein families, each represented by multiple sequence alignments and hidden Markov models (HMMs). More...

Quick Links

Sequence Search
Analyze your protein sequence for Pfam matches

View a Pfam Family
View Pfam family annotation and alignments

View a Clan
See groups of related families

View a Sequence
Look at the domain organisation of a protein sequence

View a Structure
Find the domains on a PDB structure

Keyword Search
Query Pfam by keywords

Jump To
Enter any accession or ID to jump to the page for a Pfam family or clan, UniProt sequence, PDB structure, etc.

Or view the help pages for more information

Citing Pfam

If you find Pfam useful, please consider citing the reference that describes this work:


Mirrors

The following are official Pfam mirror sites:

- WTSI, UK
- SBC, Sweden
- JFRC, USA
Family: Kinesin (PF00225)

Summary

Pfam includes annotations and additional family information from a range of different sources. These sources can be accessed via the tabs below.

Wikipedia: Kinesin  Pfam  Interpro

The Pfam group coordinates the annotation of Pfam families in Wikipedia. This family is described by a Wikipedia entry entitled "Kinesin". More...

Kinesin  Edit Wikipedia article

A kinesin is a protein belonging to a class of motor proteins found in eukaryotic cells. Kinesins move along microtubule filaments, and are powered by the hydrolysis of ATP (thus kinesins are ATPases). The active movement of kinesins supports several cellular functions including mitosis, meiosis and transport of cellular cargo, such as in axonal transport. Most kinesins walk towards the plus end of a microtubule, which, in most cells, entails transporting cargo from the centre of the cell towards the periphery. This form of transport is known as anterograde transport.

Contents

1 Structure
   1.1 Overall structure
   1.2 Kinesin motor domain
2 Cargo transport
3 Direction of motion
4 Proposed mechanisms of movement
5 Theoretical Modeling of Kinesin
6 Kinesin and mitosis
7 Family members
8 See also
9 References
10 External links
Family: **Kinesin** (PF00225)

**Domain organisation**

Below is a listing of the unique domain organisations or architectures in which this domain is found. More...

**There are 3185 sequences with the following architecture:** Kinesin

CENPE_HUMAN [Homo sapiens (Human)] Centromere-associated protein E (2701 residues)

Show all sequences with this architecture.

**There are 139 sequences with the following architecture:** Kinesin x 2

CIN8_YEAST [Saccharomyces cerevisiae (Baker's yeast)] Kinesin-like protein CIN8 (1000 residues)

Show all sequences with this architecture.

**There are 56 sequences with the following architecture:** Kinesin, FHA

KIF14_HUMAN [Homo sapiens (Human)] Kinesin-like protein KIF14 (1648 residues)

Show all sequences with this architecture.

**There are 54 sequences with the following architecture:** CH, Kinesin

Q9SS42_ARATH [Arabidopsis thaliana (Mouse-ear cress)] Kinesin-like protein (897 residues)

Show all sequences with this architecture.

**There are 54 sequences with the following architecture:** Kinesin, DUF3490

Q8LNZ2_ARATH [Arabidopsis thaliana (Mouse-ear cress)] Kinesin-like protein (938 residues)

Show all sequences with this architecture.

**There are 44 sequences with the following architecture:** Kinesin, FHA, KIF1B, DUF3694, PH

KIF1A_HUMAN [Homo sapiens (Human)] Kinesin-like protein KIF1A (1690 residues)

Show all sequences with this architecture.
Family: Kinesin (PF00225)

This family is a member of clan AAAAA (CL0023), which contains the following 157 members:

- 6PF2K
- AAA 3
- AAA 9
- Adenyl IVa2
- APS kinase
- ATP-synth_ab
- CoqE
- CTP synth_N
- DLIC
- dNK
- DUF258
- DUF853
- FekB N
- G-alpha
- Gtr1 RagA
- Herpes Helicase
- IPT
- Kinesin-relat_1
- Mg_dehlatase
- MukB
- NOG1
- PduV-EutP
- Pox A32
- Rad51
- RH3D
- SeqA DEAD
- SNF2 N
- Sulfotransfer_2
- Terminase 6
- TniB
- UvrD-helicase
- Zeta toxin
- AAA
- AAA 5
- AAA PrkA
- Adenylsucc synt
- Arch ATPase
- ATP bind 1
- CobA CobO BtuR
- Cytidylate_kin
- DNA pack_C
- DUF1253
- DUF2813
- DUF87
- Fer4_NifH
- Gal-3-0_sulfot
- Guanylate_kin
- Herpes_ori bp
- IstB IS21
- Kinesin-related
- MipZ
- MutS V
- NTPase 1
- PhoH
- PPK2
- Ras
- RHSP
- Septin
- Spore IV A
- Sulphotransf
- Terminase_GpA
- Torsin
- Viral_helicase1
- Zot
- AAA-ATPase_like
- AAA 6
- ABC ATPase
- ADK
- Arf
- ATP bind 2
- CobU
- DAP3
- DNA pack_N
- DUF1611
- DUF463
- DUF927
- Flavi DEAD
- GBP
- GvpD
- Herpes_TK
- KaC
- KTI12
- Miro
- Myosin head
- ParA
- PIF1
- PPV E1_C
- RecA
- RNA12
- Sigma54 activat
- SRP54
- T4SS-DNA transf
- Thermidylate_kin
- TraG_D C
- VirC1
- AAA 10
- AAA 7
- ABC tran
- AFG1 ATPase
- AIG1
- ArgK
- Bac DnaA
- cobW
- DEAD
- DNA_pol3_dela
- DUF2075
- DUF699
- Dynamin_N
- FTHFS
- GSP11_E
- HD2A-3
- IIGP
- KAP NTase
- LpxK
- MMR_HSR1
- NACHT
- Parvo_NS1
- Podovirus_Gp16
- PRK
- Rep_fac_C
- RNA_helicase
- SKI
- SRPRB
- Terminase_1
- TIP49
- TrwB AAD bind
- VirE
- AAA 2
- AAA 8
- ABC tran 2
- AIG
- ArsA ATPase
- CbiA
- CPT
- DEAD 2
- DnaB C
- DUF2478
- DUF815
- Exonuc_V gamma
- FtsK_SpoIIE
- GTP_EFTU
- Helicase_C
- IJPT
- Kinesin
- MCM
- MobB
- MMR
- NB-ARC
- PAXNEB
- Polyoma_ig T_C
- Rad17
- ResIII
- RuvB N
- SMC N
- Sulfortransfer 1
- UPF0079
- YhIQ
Family: Kinesin (PF00225)

Alignments

There are various ways to view or download the sequence alignments that we store. You can use a sequence viewer to look at either the seed or full alignment for the family, or you can look at a plain text version of the sequence in a variety of different formats. More...

View options

Alignment:
- Seed (87)
- Full (4150)
- NCBI (6110)
- Metagenomics (525)

Viewer:  HTML

Formatting options

Alignment:
- Seed (87)
- Full (4150)

Format:  Selex

Order:
- Tree
- Alphabetical

Sequence:
- Inserts lower case
- All upper case

Gaps:  Gaps as "." or "-" (mixed)

Download/view:
- Download
- View

Download options

Very large alignments can often cause problems for the formatting tool above. If you find that downloading or viewing a large alignment is problematic, you can also download a gzip-compressed, Stockholm-format file containing the seed or full alignment for this family.

You can also download a FASTA format file containing the full-length sequences for all sequences in the full alignment.

The main seed and full alignments are generated using sequences from the UniProt sequence database. However, we also generate
Family: Kinesin (PF00225)

HMM logo

HMM logos is one way of visualising profile HMMs. Logos provide a quick overview of the properties of an HMM in a graphical form. You can see a more detailed description of HMM logos and find out how you can interpret them [here](#). More...
**Curation and family details**

This section shows the detailed information about the Pfam family. You can see the definitions of many of the terms in this section in the glossary and a fuller explanation of the scoring system that we use in the scores section of the help pages.

### Curation

<table>
<thead>
<tr>
<th><strong>Seed source</strong></th>
<th>Prosite</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Previous IDs</strong></td>
<td>kinesin;</td>
</tr>
<tr>
<td><strong>Type</strong></td>
<td>Domain</td>
</tr>
<tr>
<td><strong>Author</strong></td>
<td>Bateman A, Finn RD</td>
</tr>
<tr>
<td><strong>Number in seed</strong></td>
<td>87</td>
</tr>
<tr>
<td><strong>Number in full</strong></td>
<td>4150</td>
</tr>
<tr>
<td><strong>Average length of the domain</strong></td>
<td>298.60 aa</td>
</tr>
<tr>
<td><strong>Average identity of full alignments</strong></td>
<td>31%</td>
</tr>
<tr>
<td><strong>Average coverage of the sequence by the domain</strong></td>
<td>34.30%</td>
</tr>
</tbody>
</table>

### HMM information

**HMM build commands:**

build method: hmmbuild -o /dev/null HMM SEED

search method: hmmsearch -Z 11384036 -E 1000 --cpu 4 HMM pfamseq

<table>
<thead>
<tr>
<th>Model details</th>
<th>Parameter</th>
<th>Sequence</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gathering cut-off</td>
<td>22.5</td>
<td>22.5</td>
<td></td>
</tr>
<tr>
<td>Trusted cut-off</td>
<td>22.5</td>
<td>22.5</td>
<td></td>
</tr>
<tr>
<td>Noise cut-off</td>
<td>22.4</td>
<td>22.4</td>
<td></td>
</tr>
</tbody>
</table>

**Model length:** 333

**Family (HMM) version:** 17
Family: Kinesin (PF00225)

Interactions

There are 6 interactions for this family. More...

- Tubulin
- Tubulin_C
- Kinesin
- Tubulin
- Kinesin
Family: Kinesin (PF00225)

Structures

For those sequences which have a structure in the Protein Data Bank®️, we use the mapping between UniProt®️, PDB and Pfam coordinate systems from the PDBera®️ group, to allow us to map Pfam domains onto UniProt sequences and three-dimensional protein structures. The table below shows the structures on which the Kinesin domain has been found.

<table>
<thead>
<tr>
<th>UniProt entry</th>
<th>UniProt residues</th>
<th>PDB ID</th>
<th>PDB chain ID</th>
<th>PDB residues</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>A8BK01_GIALA</td>
<td>11 - 335</td>
<td>2vvg</td>
<td>A</td>
<td>11 - 335</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
<td>11 - 335</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td>CENPE_HUMAN</td>
<td>12 - 329</td>
<td>1t5c</td>
<td>A</td>
<td>12 - 329</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
<td>12 - 329</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td>KAR3_YEAST</td>
<td>392 - 723</td>
<td>1f9t</td>
<td>A</td>
<td>392 - 723</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1f9u</td>
<td></td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1f9v</td>
<td></td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1f9w</td>
<td></td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3kar</td>
<td></td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td>KI13B_HUMAN</td>
<td>11 - 352</td>
<td>3gbi</td>
<td>A</td>
<td>11 - 352</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
<td>11 - 352</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td>11 - 352</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1i6</td>
<td></td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
<td>24 - 359</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1q0b</td>
<td></td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
<td>24 - 359</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1x88</td>
<td></td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td>24 - 359</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
<td>24 - 359</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td>24 - 359</td>
<td>Jmol AstexViewer SPICE</td>
</tr>
</tbody>
</table>
HMMER3: a new generation of sequence homology search software

HMMER is used for searching sequence databases for homologs of protein sequences, and for making protein sequence alignments. It implements methods using probabilistic models called profile hidden Markov models (profile HMMs).

Compared to BLAST, FASTA, and other sequence alignment and database search tools based on older scoring methodology, HMMER aims to be significantly more accurate and more able to detect remote homologs because of the strength of its underlying mathematical models. In the past, this strength came at significant computational expense, but in the new HMMER3 project, HMMER is now essentially as fast as BLAST.

As part of this evolution in the HMMER software, we are committed to making the software available to as many scientists as possible. Earlier releases of HMMER were restricted to command line use. To make the software more accessible to the wide scientific community, we now provide servers that allow sequence searches to be performed interactively via the Web.

The current version is HMMER 3.0 (28 March 2010) and can be downloaded from the software section of the site. Previous versions of the HMMER software can be obtained from the archive section.

If you have used the HMMER website, please consider citing the following reference that describes this work:

HMMER web server: interactive sequence similarity searching
R.D. Finn, J. Clements, S.R. Eddy
protein sequence vs protein sequence database

Paste in your sequence or use the example

>sp|Q14807|KIF22_HUMAN
MAAGGSTQDRRREMAAASAAASIAGACRCRSLKIGCTRPRPPPARVRVRVRLRPVFDCTGADG
SDPPCVRGMDSCSLEIANWHRHQLTQYKDFDAYRSTQCDYAIAGSVQPILRHELLEGQNN
ASVLAYGPTAGKTHHTMLGSEPQPGVPRLAMDLQQLTREGAEGEKPALSUTVMSLEYIY
QEKVLDDLDPSGDLVIREDCRGNILPLGSLKPISSFPIFERHPFPAASRNTRTVAGTRLNT
QRSSRSKHAVLLVVKDQRERLAPFRQERYKLYIDAGSEDRRTNGKGLRLESGAINTSL
LFVLGKVQDLNLQPCRVPVPDRSCLKRRDLQDSLGGSGAHSLIANIAFEPRFYLDTVSALN
FAARSKEVINRPFTNQLSLOHALGPVLKSSLQEGPAAKEARGEEIEIISGEPMAAPAPA
SASQKLSQPLKSSMDPAMLLRLLLSSLDSLASSQSGAPSLSTPKRRMVLVMTVEKDL
IEIRLTKKEQKEAKMLAQKAEEKHCPTMLRPLSRHTVGTGAKPLKAVMLPQLIQEQA
AAPNAEHIHKNCRKRKLESLDLAPEEKASEDCWELSQPSXMLAHCRQKLIDLNLNEGS
ARDLRSORIGKDAOLVCGWELHCPSEOLEDVEQGTCOMESLKLAKGLAO

Submit
Reset

Comments or questions on the site? Send a mail to hmmer@janelia.hhmi.org
Howard Hughes Medical Institute

Follow @hmmer
**HMMER**

biosequence analysis using profile hidden Markov models

---

**Job:** 9924F9AC-FEB5-11E0-A304-2B0C998A7913  
**Started:** 2011-10-24 23:01:15  
**Algorithm:** phmmer  
**HMMER Options:** -E 1 --domE 1 --incE 0.01 --incdomE 0.03 --mx BLOSUM62 --pextend 0.4 --popen 0.02 --seqdb nr

### Format

- **FASTA**
  - Download the significant hits from your search as a gzipped FASTA file.

- **Full length FASTA**
  - A gzipped file containing the full length sequences for significant search hits.

- **Aligned FASTA**
  - A gzipped file containing aligned significant search hits in FASTA format.

- **STOCKHOLM**
  - Download an alignment of significant hits as a gzipped STOCKHOLM file.

- **Text**
  - A plain text file containing the hit alignments and scores.

- **XML**
  - An XML file formatted for machine parsing of the data.

- **JSON**
  - All the results information encoded as a single json string.

- **HMM**
  - Profile HMM downloads are not available.
SUPERFAMILY Description

SUPERFAMILY is a database of structural and functional annotation for all proteins and genomes.

The SUPERFAMILY annotation is based on a collection of hidden Markov models, which represent structural protein domains at the SCOP superfamily level. A superfamily groups together domains which have an evolutionary relationship. The annotation is produced by scanning protein sequences from over 1,700 completely sequenced genomes against the hidden Markov models.

For each protein you can:
- Submit sequences for SCOP classification
- View domain organisation, sequence alignments and protein sequence details

For each genome you can:
- Examine superfamiliy assignments, phylogenetic trees, domain organisation lists and networks
- Check for over- and under-represented superfamilies within a genome

For each superfamily you can:
- Inspect SCOP classification, functional annotation, Gene Ontology annotation, InterPro abstract and genome assignments
- Explore taxonomic distribution of a superfamily across the tree of life

All annotation, models and the database dump are freely available for download to everyone.
Keyword Search Results

Results 1-3 of 3 for kinesin.

1.

**SCOP classification**
- **Class**: Alpha and beta proteins (a/b)
- **Fold**: P-loop containing nucleoside triphosphate hydrolases
- **Superfamily**: P-loop containing nucleoside triphosphate hydrolases
- **Family**: Motor proteins
- **Protein**: Kinesin
- **Protein**: Kinesin motor Ncd (non-claret disjunctual)
- **Protein**: Kinesin heavy chain-like protein
Motor proteins family

**SCOP classification**
- Root: [SCOP hierarchy in SUPERFAMILY](http://supfam.cs.bris.ac.uk/SUPERFAMILY/cgi-bin/scop.cgi?sunid=52641) (11)
- Class: [Alpha and beta proteins (a/b)](http://supfam.cs.bris.ac.uk/SUPERFAMILY/cgi-bin/scop.cgi?sunid=51349) (147)
- Fold: [P-loop containing nucleoside triphosphate hydrolases](http://supfam.cs.bris.ac.uk/SUPERFAMILY/cgi-bin/scop.cgi?sunid=52539)
- Superfamily: [P-loop containing nucleoside triphosphate hydrolases](http://supfam.cs.bris.ac.uk/SUPERFAMILY/cgi-bin/scop.cgi?sunid=52540) (24)
- Family: [Motor proteins](http://supfam.cs.bris.ac.uk/SUPERFAMILY/cgi-bin/scop.cgi?sunid=52641) (4)

**Family statistics**

<table>
<thead>
<tr>
<th>Genomes</th>
<th>Uniprot 2011_09</th>
<th>PDB chains (SCOP 1.75)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domains</td>
<td>15,002</td>
<td>10,025</td>
</tr>
<tr>
<td>Proteins</td>
<td>14,811</td>
<td>9,977</td>
</tr>
</tbody>
</table>

**Gene Ontology (high-coverage)**

<table>
<thead>
<tr>
<th>GO term</th>
<th>FDR (all)</th>
<th>SDF0 level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biological Process (BP)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>multicellular organismal process</td>
<td>0</td>
<td>Least Inf</td>
</tr>
<tr>
<td>biological regulation</td>
<td>0.0357</td>
<td>Least Inf</td>
</tr>
</tbody>
</table>
P-loop containing nucleoside triphosphate hydrolases superfamily

**SCOP classification**

- **Root:** SCOP hierarchy in SUPERFAMILY [SCOP 0] (11)
- **Class:** Alpha and beta proteins (a/b) [SCOP 51349] (147)
- **Fold:** P-loop containing nucleoside triphosphate hydrolases [SCOP 52539]
- **Superfamily:** P-loop containing nucleoside triphosphate hydrolases [SCOP 52540] (24)
- **Families:** Nucleotide and nucleoside kinases [SCOP 52541] (20)
  - Shikimate kinase (AroK) [SCOP 52566]
  - Chloramphenicol phosphotransferase [SCOP 52569]
  - Gluconate kinase [SCOP 75195]
  - Plasmid maintenance system epsilon/zeta, toxin zeta subunit [SCOP 82395]
  - Adenosine-5'phosphosulfate kinase (APS kinase) [SCOP 52572]
  - ATP sulfurylase C-terminal domain [SCOP 64011]
  - PAPS sulfotransferase [SCOP 52575] (14)
  - Phosphoribulokinase/pantothenate kinase [SCOP 52584] (5)
  - 6-phosphofructo-2-kinase/fructose-2,6-bisphosphatase, kinase domain [SCOP 52589]
  - G proteins [SCOP 52592] (78)
  - Motor proteins [SCOP 52641] (4)
  - Nitrogenase iron protein-like [SCOP 52652] (15)
  - RecA protein-like (ATPase-domain) [SCOP 52670] (17)
  - Bacterial cell division inhibitor SulA [SCOP 89678]
  - ABC transporter ATPase domain-like [SCOP 52686] (23)
  - Tandem AAA-ATPase domain [SCOP 81268] (23)
  - Extended AAA-ATPase domain [SCOP 81269] (23)
  - RNA helicase [SCOP 52724] (3)
  - Helicase-like "domain" of reverse gyrase [SCOP 69496]
  - DNA helicase UvsW [SCOP 102396]
  - YjieE-like [SCOP 75213]
  - Type II thymidine kinase [SCOP 117558]
That’s it!