Hidden Markov Models, HMM’s

Morten Nielsen
Department of Bio and Health Informatics, DTU
Objectives

- Introduce Hidden Markov models and understand that they are just weight matrices with gaps
- How to construct an HMM
- How to “align/score” sequences to HMM’s
  - Viterbi decoding
  - Forward decoding
  - Backward decoding
  - Posterior Decoding
- Use and construct a Profile HMM
  - HMMer
Markov Chains

**States**: Three states - sunny, cloudy, rainy.

**State transition matrix**: The probability of the weather given the previous day's weather.

\[
\begin{pmatrix}
\text{Sun} & \text{Cloud} & \text{Rain} \\
0.5 & 0.25 & 0.25 \\
0.375 & 0.125 & 0.375 \\
0.125 & 0.625 & 0.375 \\
\end{pmatrix}
\]

**Initial Distribution**: Defining the probability of the system being in each of the states at time 0.
Hidden Markov Models

**Hidden states**: the (TRUE) states of a system that may be described by a Markov process (e.g., the weather).

**Observable states**: the states of the process that are `visible' (e.g., seaweed dampness).

---

Emission probabilities

Transition probabilities
TMHMM (trans-membrane HMM) (Sonnhammer, von Heijne, and Krogh)

Extra cellular

Trans membrane

Intra cellular
TMHMM (trans-membrane HMM)  
(Sonnhammer, von Heijne, and Krogh)

A  
[Diagram showing membrane elements and their interactions]

B  
[Diagram showing helix core with numbers 1 to 25]

C  
[Diagram showing loop and cap elements]

Model TM length distribution.  
Power of HMM.  
Difficult in alignment.

ALLYVDWQILPVIL
Gene finding

Start codon: CGCCATG
5' UTR: CCTCCCAG
3' UTR: GATCCC
Intron: Removed
Promoter
Exon (Left)
Acceptor site: CCTG
Poly-A site: ACCAC
Stop codon: TGA
Donor site: GTGAGT

Transcription start
Weight matrix construction

SLLPAIVEL YLLPAIVHI TLWVDPYEV GLVPFLSVS KLLEPVLLL LLDVPTAAT LLDVPTAAT LLDVPTAAT
LLDVPTAAT VLFRGGPFRG MVGDTRLLL YMNQTMQSGL MLLSVPLL SLGLLVEV ALLPPINIL TLIKIQLTTL
HLIDLYTVS ILAPPVVKL ALPQVVLIL GILGFVTIL STRTHQSGQ CGLDVLTAQ KIRLAVKK QVCRPIPTI
ILFGHENRNV ILMEHIHKL ILDQKINEV SLAGIGITV LLIENWAVSL FLLWATAEA SLPDGQISY KKKEEAPSRL
LERPQGNEI ALSNEVVKL ALNLYQHVL DLERKVESL FLQGENSNLF ALSQDHYL GLESEFTYEL STAPPAGV
PLDEGYFTL GVLVGVALI RLLDQKVELV HLSTAFARV RLDSYVRLS YMNQTMQV GILGFVTLL ILKEPQHVGL
ILGFVFTL LLFGPYNVVL GLSPTVVLQ WLLAVPFTVE FLPSDFSPS CLGLDLTMV FIAGNSAYE KLGKEFYNQM
KLVAGANIA DLGMYQPLV RLVTLKDIV MVVAVYCL AAGIGILTV YLEPGVPVA LLDGATER ITDQVPSVF
KTVGQYQVQ LTTDQVPFSS APEHVAREL YLDKIQNSL MRRKLLAIS AIMKDNNIIK IMDKDNNLKM SICGNWAKV
SLIPAGAKQ KACFSIALFL ELVSEFSRM KLTPLCVTL VLLRGPSFSL YIGEVLTVS QIEVCSWTQ VMNQVLQY
ILTVLGPLV KLVELYIKV FLWGPALV GLSRYVARL FLTRITLT HGLNWYVL GIAGGLALL GLQCTMLYV
TGAPVYQST VYIQMDLVL VLPDIFIRC VLPDIFIRC AAGIGIAYV LVLQGLLAV ALGGLPLFV GIGIGILAA
GAGIGVAVL IAGIGILAI LIVIGILIL LAGIGILIA IVDIGIGLIT IAGIGILTV AAIGIIQI QAGIGILLAA
KARDPHSGH KACFSIALFL SLNTWYTAL GRQGTAFVTL NLVMQVATQ GLSCFQVLOVL PQKQHFLQ
AVFDRKSDA LLDFVRFMG VLVKSPHNL GLAPPQHOL LLLGRNQFEL PTLFGWCYK LEWRFDSRR TLNAWVKVV
GLCITVAML FDSYCVQV IASAVVGLY VMAGVFSPY LLWTVVLVL SVRDLRLV LMMDCSGL LCLSTVQLV
VLHDDLEA LMDITQFLQ SSLMWWTQ QLSLMLWT LLATCMFV RLTFRQSRV YMDSQMSQV FLLPKLQC
ISNDVCAQV VKTDGPHPYE FVTLQVHHL FLGQVALLA VLLFNDFTFL LMWAKIQFV NLLLEAVE LSRSWSG
YTAFTIPSJ RLMQDFSLR LRLDPIFCSL FWLPGRAWV RLLQETELV SLPFEGIDFY SLDQSVVSL RLMNFTFYI
NMFETFYIG LMIQILPNLV TLFGSHSVV SLTVTTTV VLGASLVF IAKLFHLWL SAPPJMVNL LLLLTVTLV
VVLGQVFJG LHHGAGSLI MLQKCMQRI MLGHTMEV MLGHTMEV SLADTNLSA LLRAWPRPL CVLQCTMQK
GLYDGMEHJ KLMLVHFLD QLQFVGSFIE MLAAQEEAL IMAAQEALF VYDQHETTV YLSGARLNL RMFSPAN
EAAGIGILT TLDSQQMSL TSSPPGFTRV KVAELHFL IMIGVLTGVG ALCRWQGLL LLFAGQVQF QLLCESTAV
YLSTAFARV YLLEMLWRL SLDDYHNLV RLDKQVLEV GILFQVLYQV KLQANTRV FYIAGLSSA KLVANNTVL
FLDFMIGEV ALQGTVALL VLQGDLVLL SLYDSPPEE ALYVDSLFF SLQQQLIGL EILTGEFLQ MINALYDLKL
AAGIGILTV FLPSDFSPS SVRDRLRL SREWLLRI LLSAWIITA AAGIGILTV AVPDEIPPL FAYDGDYI
AAGIGILTV FLPSDFSPS AAGIGILTV FLPSDFSPS AAGIGILTV FWGPRALV ETVSEQSNV LTLWQRLPLV
PSSM construction

- Calculate amino acid frequencies at each position using
  - Sequence weighting
  - Pseudo counts
- Define background model
  - Use background amino acids frequencies
- PSSM is

\[ S(a_i) = \log \frac{p(a_i)}{q(a)} \]
More on scoring

\[ S = \sum_i S(a_i) \]

Probability of observation given Model

\[ S = \log \left( \frac{\prod p(a_i)}{\prod q(a_i)} \right) \]

Probability of observation given Prior (background)

\[ S = \log \left( \frac{P(a \mid M)}{P(a \mid B)} \right) \]
Hidden Markov Models

• Weight matrices do not deal with insertions and deletions
• In alignments, this is done in an ad-hoc manner by optimization of the two gap penalties for first gap and gap extension
• HMM is a natural framework where insertions/deletions are dealt with explicitly
## Multiple sequence alignment

### Learning from evolution

<table>
<thead>
<tr>
<th>QUERY</th>
<th>HAMDIRCYHSGG-PLHL-GEI-EDFNGQSCIVCPWHKYKITLATGE-GLYQSINPKDPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q8K2P6</td>
<td>HAMDIRCYHSGG-PLHL-GEI-EDFNGQSCIVCPWHKYKITLATGE-GLYQSINPKDPS</td>
</tr>
<tr>
<td>Q8TAC1</td>
<td>HAMDIRCYHSGG-PLHL-GEI-EDFNGQSCIVCPWHKYKITLATGE-GLYQSINPKDPS</td>
</tr>
<tr>
<td>Q07947</td>
<td>FAVQDTCTHGWD-ALSD-GYL-DGD----IVECTLHFGKFCVRTGK-VKAL------PA</td>
</tr>
<tr>
<td>P0A185</td>
<td>YATDNLCTHGSA-RMSE-GYI-EGRE----IECPLHGRFDVCTGK-VKAL------PA</td>
</tr>
<tr>
<td>P0A186</td>
<td>YATDNLCTHGSA-RMSE-GYI-EGRE----IECPLHGRFDVCTGK-VKAL------PA</td>
</tr>
<tr>
<td>Q51493</td>
<td>YATDNLCTHGA-RMSE-GFI-EGRE----IECPLHGRFDVCTGGR-VKAL------PA</td>
</tr>
<tr>
<td>A5W4F0</td>
<td>FAVQDTCTHGWD-ALSD-GYI-OGD----IVECTLHFGKFCVRTGK-VKAL------PA</td>
</tr>
<tr>
<td>P0C620</td>
<td>FAVQDTCTHGWD-ALSD-GYI-OGD----IVECTLHFGKFCVRTGK-VKAL------PA</td>
</tr>
<tr>
<td>P08086</td>
<td>FAVQDTCTHGWD-ALSD-GYI-OGD----IVECTLHFGKFCVRTGK-VKAL------PA</td>
</tr>
<tr>
<td>Q52440</td>
<td>FATQDCCTHGEW-SLSE-GGY-LDGD----VVECSLHMGKFCVRTGK-----------V</td>
</tr>
<tr>
<td>Q7N4V8</td>
<td>FAVDDRCSHGNA-SISE-GYL-ED-----NATVECPLHTASFCLRTGK-ALCL------PA</td>
</tr>
<tr>
<td>P37332</td>
<td>FATQDCCTHGWD-SLSD-GYI-EGD----VVECSLHMGKFCVRTGK-----------V</td>
</tr>
<tr>
<td>A7ZPY3</td>
<td>YAINDRCSHGNA-SMSE-GYI-EDDD-----ATVECPHLAASFCLKTGK-ALCL------PA</td>
</tr>
<tr>
<td>P0ABW1</td>
<td>YAINDRCSHGNA-SMSE-GYI-EDDD-----ATVECPHLAASFCLKTGK-ALCL------PA</td>
</tr>
<tr>
<td>A8A346</td>
<td>YAINDRCSHGNA-SMSE-GYI-EDDD-----ATVECPHLAASFCLKTGK-ALCL------PA</td>
</tr>
<tr>
<td>P0ABW0</td>
<td>YAINDRCSHGNA-SMSE-GYI-EDDD-----ATVECPHLAASFCLKTGK-ALCL------PA</td>
</tr>
<tr>
<td>P0ABW2</td>
<td>YAINDRCSHGNA-SMSE-GYI-EDDD-----ATVECPHLAASFCLKTGK-ALCL------PA</td>
</tr>
<tr>
<td>Q3YZ13</td>
<td>YAINDRCSHGNA-SMSE-GYI-EDDD-----ATVECPHLAASFCLKTGK-ALCL------PA</td>
</tr>
<tr>
<td>Q06458</td>
<td>YALDNLEPGSEANVLSR-GLL-GDAGGEPIVISPLYKQRIRLDG-----------V</td>
</tr>
</tbody>
</table>

**Legend:**
- **Core**
- **Insertion**
- **Deletion**

**Purpose:**
- Core: Conserved positions across all sequences.
- Insertion: Positions present in some sequences but not in others.
- Deletion: Positions missing in some sequences compared to others.
Why hidden?

The unfair casino: Loaded die $p(6) = 0.5$; switch fair to load: 0.05; switch load to fair: 0.1

- Model generates numbers
  - 312453666641
- Does not tell which die was used
- Alignment (decoding) can give the most probable solution/path (Viterbi)
  - FFFFFFLLLLLL
- Or most probable set of states
  - FFFFFFLLLLLL
HMM (a simple example)

Example from A. Krogh
Core region defines the number of states in the HMM (red)
Insertion and deletion statistics are derived from the non-core part of the alignment (black)
HMM construction (supervised learning)

- 5 matches. A, 2xC, T, G
- 5 transitions in gap region
  - C out, G out
  - A-C, C-T, T out
  - Out transition 3/5
  - Stay transition 2/5

\[
\text{ACA---ATG} \quad 0.8 \times 0.8 \times 0.8 \times 0.4 \times 1 \times 0.8 \times 1 \times 0.2 = 3.3 \times 10^{-2}
\]
Scoring a sequence to an HMM

\[
\begin{align*}
\text{ACA---ATG} & : 0.8 \times 1 \times 0.8 \times 1 \times 0.8 \times 0.4 \times 1 \times 0.8 \times 1 \times 0.2 = 3.3 \times 10^{-2} \\
\text{TCAACTATC} & : 0.2 \times 1 \times 0.8 \times 1 \times 0.8 \times 0.6 \times 0.2 \times 0.4 \times 0.4 \times 0.4 \times 0.2 \times 0.6 \times 1 \times 1 \times 0.8 \times 1 \times 0.8 = 0.0075 \times 10^{-2} \\
\text{ACAC--AGC} & : 1.2 \times 10^{-2}
\end{align*}
\]

Consensus:
\[
\text{ACAC--ATC} = 4.7 \times 10^{-2}, \quad \text{ACA---ATC} = 13.1 \times 10^{-2}
\]

Exceptional:
\[
\text{TGCT--AGG} = 0.0023 \times 10^{-2}
\]
Align sequence to HMM - Null model

- Score depends **strongly** on length
- Null model is a random model. For length $L$ the score is $0.25^L$
- Log-odds score for sequence $S$
  \[
  \log\left( \frac{P(S)}{0.25^L} \right)
  \]
- Positive score means more likely than Null model

\[
\begin{align*}
  \text{ACA---ATG} &= 4.9 \\
  \text{TCAACTATC} &= 3.0 \\
  \text{ACAC--AGC} &= 5.3 \\
  \text{AGA---ATC} &= 4.9 \\
  \text{ACCG--ATC} &= 4.6 \\
  \text{Consensus: ACAC--ATC} &= 6.7 \\
  \text{ACA---ATC} &= 6.3 \\
  \text{Exceptional: TGCT--AGG} &= -0.97
\end{align*}
\]

- This is just like we did for PSSM log(p/q)!

*Note!*
Aligning a sequence to an HMM

- Find the path through the HMM states that has the highest probability
  - For alignments, we found the path through the scoring matrix that had the highest sum of scores
- The number of possible paths rapidly gets very large making brute force search infeasible
  - Just like checking all path for alignment did not work
- Use dynamic programming
  - The Viterbi algorithm does the job
The Viterbi algorithm

- Model generates numbers - 312453666641

*The unfair casino*: Loaded dice \( p(6) = 0.5 \); switch fair to load: 0.05; switch load to fair: 0.1

![Diagram showing states and transitions for the Viterbi algorithm with numbers and probabilities.]
Model decoding (Viterbi)

- Example: 566. What was the most likely series of dice used to generate this output?
- Use Brute force

<table>
<thead>
<tr>
<th>Dice 1</th>
<th>Dice 2</th>
<th>Dice 3</th>
<th>Dice 4</th>
<th>Dice 5</th>
<th>Dice 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
</tr>
<tr>
<td>1/10</td>
<td>1/10</td>
<td>1/10</td>
<td>1/10</td>
<td>1/10</td>
<td>1/2</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\text{FFF} &= 0.5 \times 0.167 \times 0.95 \times 0.167 \times 0.95 \times 0.167 = 0.0021 \\
\text{FFL} &= 0.5 \times 0.167 \times 0.95 \times 0.167 \times 0.05 \times 0.5 = 0.00333 \\
\text{FLF} &= 0.5 \times 0.167 \times 0.05 \times 0.5 \times 0.1 \times 0.167 = 0.000035 \\
\text{FLL} &= 0.5 \times 0.167 \times 0.05 \times 0.5 \times 0.9 \times 0.5 = 0.00094 \\
\text{LFF} &= 0.5 \times 0.1 \times 0.1 \times 0.167 \times 0.95 \times 0.167 = 0.00013 \\
\text{LFL} &= 0.5 \times 0.1 \times 0.1 \times 0.167 \times 0.05 \times 0.5 = 0.000021 \\
\text{LLF} &= 0.5 \times 0.1 \times 0.9 \times 0.5 \times 0.1 \times 0.167 = 0.00038 \\
\text{LLL} &= 0.5 \times 0.1 \times 0.9 \times 0.5 \times 0.9 \times 0.5 = 0.0101
\end{align*}
\]
Or in log space

- Example: 566. What was the most likely series of dice used to generate this output?

\[
\log(P(LLL|M)) = \log(0.5 \times 0.1 \times 0.9 \times 0.5 \times 0.9 \times 0.5) = \log(0.0101)
\]

or

\[
\log(P(LLL|M)) = \log(0.5) + \log(0.1) + \log(0.9) + \log(0.5) + \log(0.9) + \log(0.5) \\
= -0.3 -1 -0.046 -0.3 -0.046 -0.3 = -1.99
\]
Model decoding (Viterbi)

- Example: 566611234. What was the most likely series of dice used to generate this output?

\[
F = 0.5 \times 0.167 \\
\log(F) = \log(0.5) + \log(0.167) = -1.08 \\
L = 0.5 \times 0.1 \\
\log(L) = \log(0.5) + \log(0.1) = -1.30
\]

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>F</strong></td>
<td>-1.08</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>L</strong></td>
<td>-1.30</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Log model:

- 1: -0.78
- 2: -0.78
- 3: -0.78
- 4: -0.78
- 5: -0.78
- 6: -0.78

Fair:

- 1: -1
- 2: -1
- 3: -1
- 4: -1
- 5: -1
- 6: -0.3

Loaded:

- 1: -1
- 2: -1
- 3: -1
- 4: -1
- 5: -1
- 6: -0.3
Model decoding (Viterbi)

- Example: 566611234. What was the most likely series of dice used to generate this output?

\[
\begin{align*}
FF &= 0.5 \times 0.167 \times 0.95 \times 0.167 \\
\log(FF) &= -0.30 - 0.78 - 0.02 - 0.78 = -1.88 \\
LF &= 0.5 \times 0.1 \times 0.1 \times 0.167 \\
\log(LF) &= -0.30 - 1 - 1 - 0.78 = -3.08 \\
FL &= 0.5 \times 0.167 \times 0.05 \times 0.5 \\
\log(FL) &= -0.30 - 0.78 - 1.30 - 0.30 = -2.68 \\
LL &= 0.5 \times 0.1 \times 0.9 \times 0.5 \\
\log(LL) &= -0.30 - 1 - 0.046 - 0.3 = -1.65
\end{align*}
\]

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1.08</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1.30</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fair

-1

Loaded

-1

1: -0.78
2: -0.78
3: -0.78
4: -0.78
5: -0.78
6: -0.78

-0.02

Log model

-1.3

-0.046
Model decoding (Viterbi)

- Example: 566611234. What was the most likely series of dice used to generate this output?

\[
\begin{align*}
FF &= 0.5 \times 0.167 \times 0.95 \times 0.167 \\
\text{Log}(FF) &= -0.30 -0.78 -0.02 -0.78 = -1.88 \\
LF &= 0.5 \times 0.1 \times 0.1 \times 0.167 \\
\text{Log}(LF) &= -0.30 -1 -1 -0.78 = -3.08 \\
FL &= 0.5 \times 0.167 \times 0.05 \times 0.5 \\
\text{Log}(FL) &= -0.30 -0.78 -1.30 -0.30 = -2.68 \\
LL &= 0.5 \times 0.1 \times 0.9 \times 0.5 \\
\text{Log}(LL) &= -0.30 -1 -0.046 -0.3 = -1.65
\end{align*}
\]

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>-1.08</td>
<td>-1.88</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Model decoding (Viterbi)

- Example: 566611234. What was the most likely series of dice used to generate this output?

---

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>-1.08</td>
<td>-1.88</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

\[
\begin{align*}
\text{FFF} &= 0.5 \times 0.167 \times 0.95 \times 0.167 \times 0.95 \times 0.167 = 0.0021 \\
\text{FLF} &= 0.5 \times 0.167 \times 0.05 \times 0.5 \times 0.1 \times 0.167 = 0.00035 \\
\text{LFF} &= 0.5 \times 0.1 \times 0.1 \times 0.167 \times 0.95 \times 0.167 = 0.00013 \\
\text{LLF} &= 0.5 \times 0.1 \times 0.9 \times 0.5 \times 0.1 \times 0.167 = 0.00038 \\
\text{FLL} &= 0.5 \times 0.167 \times 0.05 \times 0.9 \times 0.5 \times 0.5 = 0.00094 \\
\text{FFL} &= 0.5 \times 0.167 \times 0.95 \times 0.167 \times 0.05 \times 0.5 = 0.00333 \\
\text{LFL} &= 0.5 \times 0.1 \times 0.1 \times 0.167 \times 0.05 \times 0.5 = 0.000021 \\
\text{LLL} &= 0.5 \times 0.1 \times 0.9 \times 0.5 \times 0.9 \times 0.5 = 0.0101
\end{align*}
\]
Model decoding (Viterbi)

- Example: 566611234. What was the most likely series of dice used to generate this output?

\[
\begin{align*}
\text{FFF} &= 0.5 \times 0.167 \times 0.95 \times 0.167 \times 0.95 \times 0.167 = 0.0021 \\
\log(P(\text{FFF})) &= -2.68 \\
\text{LLL} &= 0.5 \times 0.1 \times 0.9 \times 0.5 \times 0.9 \times 0.5 = 0.0101 \\
\log(P(\text{LLL})) &= -1.99
\end{align*}
\]

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>-1.08</td>
<td>-1.88</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>-0.78</td>
<td>-0.78</td>
<td>-0.78</td>
<td>-0.78</td>
</tr>
<tr>
<td>L</td>
<td>-1.3</td>
<td>-1</td>
<td>-1</td>
<td>-0.3</td>
</tr>
</tbody>
</table>

Log model

-0.02 -1.3 -1 -0.046
Model decoding (Viterbi)

- Example: 566611234. What was the most likely series of dice used to generate this output?

\[
\begin{align*}
\text{FFF} &= 0.5 \times 0.167 \times 0.95 \times 0.167 \times 0.95 \times 0.167 = 0.0021 \\
\log(P(\text{FFF})) &= -2.68 \\
\text{LLL} &= 0.5 \times 0.1 \times 0.9 \times 0.5 \times 0.9 \times 0.5 = 0.0101 \\
\log(P(\text{LLL})) &= -1.99
\end{align*}
\]

<table>
<thead>
<tr>
<th></th>
<th>F</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>-1.08</td>
<td>-1.88</td>
<td>-2.68</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td>-1.99</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Log model:

- F: -2.68
- L: -1.99
Model decoding (Viterbi)

- Example: 566611234. What was the most likely series of dice used to generate this output?

\[-1.88 - 0.02 - 0.78 = -2.68\]

\[-1.65 - 1 - 0.78 = -3.43\]

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>-1.08</td>
<td>-1.88</td>
<td>-2.68</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td>-1.99</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Model decoding (Viterbi)

- Example: 566611234. What was the most likely series of dice used to generate this output?

\[-1.88 - 1.3 - 0.3 = -3.48\]

\[-1.65 - 0.046 - 0.3 = -1.99\]

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>-1.08</td>
<td>-1.88</td>
<td>-2.68</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td>-1.99</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Log model

- Fair
  - 1: -0.78
  - 2: -0.78
  - 3: -0.78
  - 4: -0.78
  - 5: -0.78
  - 6: -0.78

- Loaded
  - 1: -1
  - 2: -1
  - 3: -1
  - 4: -1
  - 5: -1
  - 6: -0.3

\[\log_{10}(0) = -0.02\]

\[\log_{10}(0) = -1.08\]

\[\log_{10}(0) = -1.30\]
Model decoding (Viterbi)

- Example: 566611234. What was the most likely series of dice used to generate this output?

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>-1.08</td>
<td>-1.88</td>
<td>-2.68</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td>-1.99</td>
<td></td>
</tr>
</tbody>
</table>
Model decoding (Viterbi)

• Example: 566611234. What was the most likely series of dice used to generate this output?

\[
-0.78 - 0.02 - 2.68 = -3.48
\]

\[
-0.78 - 1 - 1.99 = -3.77
\]
Model decoding (Viterbi)

- Example: 566611234. What was the most likely series of dice used to generate this output?

\[-0.78 - 0.02 - 2.68 = -3.48\]
\[-0.78 - 1 - 1.99 = -3.77\]

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>-1.08</td>
<td>-1.88</td>
<td>-2.68</td>
<td>-3.48</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td>-1.99</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Model decoding (Viterbi)

- Now we can formalize the algorithm!

\[ P_i(i + 1) = p_i(i + 1) \cdot \max_k (P_k(i) \cdot a_{kl}) \quad \text{or} \]

\[ \log(P_i(i + 1)) = \log(p_i(i + 1)) + \max_k (\log(P_k(i)) + \log(a_{kl})) \]
Model decoding (Viterbi). Can you do it?

- Example: 566611234. What was the most likely series of dice used to generate this output?

- Fill out the table using the Viterbi recursive algorithm
  - Add the arrows for backtracking

- Find the optimal path

\[
P_i(i + 1) = p_i(i + 1) \cdot \max_k (P_k(i) \cdot a_{kl}) \quad \text{or} \\
\log(P_i(i + 1) = \log(p_i(i + 1)) + \max_k (\log(P_k(i)) + \log(a_{kl})))
\]

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>-1.08</td>
<td>-1.88</td>
<td>-2.68</td>
<td>-3.48</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td>-1.99</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Model decoding (Viterbi). Can you do it?

- Example: 566611234. What was the most likely series of dice used to generate this output?
- Fill out the table using the Viterbi recursive algorithm
  - Add the arrows for backtracking
- Find the optimal path

\[
P_i(i+1) = p_i(i+1) \cdot \max_k(P_k(i) \cdot a_{kl}) \quad \text{or} \quad \log(P_i(i+1) = \log(p_i(i+1)) + \max_k(\log(P_k(i) + \log(a_{kl})))
\]

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-1.08</td>
<td>-1.88</td>
<td>-2.68</td>
<td>-3.48</td>
<td>-4.92</td>
<td></td>
<td></td>
<td>-6.52</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td>-1.99</td>
<td>-3.39</td>
<td></td>
<td></td>
<td></td>
<td>-6.53</td>
<td></td>
</tr>
</tbody>
</table>

Log model

\[1: -0.78\]
\[2: -0.78\]
\[3: -0.78\]
\[4: -0.78\]
\[5: -0.78\]
\[6: -0.78\]

Fair

\[1: -1\]
\[2: -1\]
\[3: -1\]
\[4: -1\]
\[5: -1\]
\[6: -0.3\]

Loaded

\[-0.02\]
\[-1.3\]
\[-1\]
\[-0.046\]
Model decoding (Viterbi). Can you do it?

- Example: 566611234. What was the most likely series of dice used to generate this output?
- Fill out the table using the Viterbi recursive algorithm
  - Add the arrows for backtracking
- Find the optimal path

\[
P_i(i+1) = p_i(i+1) \cdot \max_k (P_k(i) \cdot a_{kl}) \quad \text{or} \quad 
\log(P_i(i+1)) = \log(p_i(i+1)) + \max_k (\log(P_k(i)) + \log(a_{kl}))
\]

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>-1.08</td>
<td>-1.88</td>
<td>-2.68</td>
<td>-3.48</td>
<td>-4.12</td>
<td>-4.92</td>
<td>-5.72</td>
<td>-6.52</td>
<td>-7.33</td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td>-1.99</td>
<td>-2.34</td>
<td>-3.39</td>
<td>-4.44</td>
<td>-5.49</td>
<td>-6.53</td>
<td>-7.57</td>
</tr>
</tbody>
</table>
Model decoding (Viterbi). Can you do it?

- Example: 566611234. What was the most likely series of dice used to generate this output?
- The most likely path is - LLLLLFFF

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>-1.08</td>
<td>-1.88</td>
<td>-2.68</td>
<td>-3.48</td>
<td>-4.12</td>
<td>-4.92</td>
<td>-5.72</td>
<td>-6.52</td>
<td>-7.33</td>
</tr>
<tr>
<td>L</td>
<td>-1.30</td>
<td>-1.65</td>
<td>-1.99</td>
<td>-2.34</td>
<td>-3.39</td>
<td>-4.44</td>
<td>-5.49</td>
<td>-6.53</td>
<td>-7.57</td>
</tr>
</tbody>
</table>

Log model:
- Fair:
  - 1: -0.78
  - 2: -0.78
  - 3: -0.78
  - 4: -0.78
  - 5: -0.78
  - 6: -0.78
- Loaded:
  - 1: -1
  - 2: -1
  - 3: -1
  - 4: -1
  - 5: -1
  - 6: -0.3
Model decoding (Viterbi).

- What happens if you have three dice?

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L1</td>
<td></td>
<td></td>
<td></td>
<td>-1.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L2</td>
<td></td>
<td></td>
<td>-1.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
P_i(i+1) = p_i(i+1) \cdot \max_k (P_k(i) \cdot a_{kl}) \text{ or } \\
\log(P_i(i+1)) = \log(p_i(i+1)) + \max_k (\log(P_k(i)) + \log(a_{kl}))
\]
And if you have a trans-membrane model

- What is the most likely path (alignment) of a protein sequence to the model

<table>
<thead>
<tr>
<th></th>
<th>D</th>
<th>G</th>
<th>V</th>
<th>L</th>
<th>I</th>
<th>M</th>
<th>A</th>
<th>D</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>iC</td>
<td>-1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>-1.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>xC</td>
<td>-1.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
P_i(i + 1) = p_i(i + 1) \cdot \max_k (P_k(i) \cdot a_{kl}) \quad \text{or} \quad \\
\log(P_i(i + 1)) = \log(p_i(i + 1)) + \max_k (\log(P_k(i)) + \log(a_{kl}))
\]
The Forward algorithm

• The Viterbi algorithm finds the most probable path giving rise to a given sequence

• One other interesting question would be
  - What is the probability that a given sequence can be generated by the hidden Markov model
    • Calculated by summing over all path giving rise to a given sequence
The Forward algorithm

- Calculate summed probability over all paths giving rise to a given sequence

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

- The number of possible paths is very large making (once more) brute force calculations infeasible
  - Use dynamic (recursive) programming
The Forward algorithm

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

- Say we know the probability of generating the sequence up to and including \( x_i \) ending in state \( k \)

\[ f_k(i) = P(x_1, x_2, \ldots, x_i, \pi_i = k) \]

- Then the probability of observing the element \( i+1 \) of \( x \) ending in state \( l \) is

\[ f_l(i + 1) = p_l(x_{i+1}) \cdot \sum_{k} f_k(i) \cdot a_{kl} \]

- where \( p_l(x_{i+1}) \) is the probability of observing \( x_{i+1} \) is state \( l \), and \( a_{kl} \) is the transition probability from state \( k \) to state \( l \)

- Then

\[ P(x) = \sum_{k} f_k(L) \]
Forward algorithm

\[ f_l(i + 1) = p_l(x_{i+1}) \cdot \sum_{k} f_k(i) \cdot a_{kl} \]

\[ f_k(0) = 1 \]

\[ a_{0l} = \pi_l \]

\[ f_F(5) = 0.167 \cdot 0.5 = 0.083 \]

\[ f_L(5) = 0.1 \cdot 0.5 = 0.05 \]
Forward algorithm

\[ f_l(i+1) = p_l(x_{i+1}) \cdot \sum_k f_k(i) \cdot a_{kl} \]

\[ f_k(0) = 1 \]

\[ a_{0l} = \pi_l \]

\[ f_F(5) = 0.167 \cdot 0.5 = 0.083 \]

\[ f_L(5) = 0.1 \cdot 0.5 = 0.05 \]
Forward algorithm

\[ f_{l}(i+1) = p_{l}(x_{i+1}) \cdot \sum_{k} f_{k}(i) \cdot a_{kl} \]
Forward algorithm

\[ f_l(i + 1) = p_l(x_{i+1}) \cdot \sum_k f_k(i) \cdot a_{kl} \]

\[ 0.167 \cdot (0.083 \cdot 0.95 + 0.05 \cdot 0.1) = 0.014 \]
Forward algorithm

\[ f_l(i + 1) = p_l(x_{i+1}) \cdot \sum_k f_k(i) \cdot a_{kl} \]

0.167 \cdot (0.083 \cdot 0.95 + 0.05 \cdot 0.1) = 0.014
Forward algorithm. Can you do it yourself?

\[ f_l(i + 1) = p_l(x_{i+1}) \cdot \sum_k f_k(i) \cdot a_{kl} \]

Fill out the empty cells in the table!
What is \( P(x) \)?

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>8.30e-2</td>
<td>2.63e-3</td>
<td>6.08e-4</td>
<td>1.82e-4</td>
<td>3.66e-5</td>
<td>1.09e-6</td>
<td>1.79e-7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>5.00e-2</td>
<td>2.46e-2</td>
<td>1.14e-2</td>
<td>4.71e-4</td>
<td>4.33e-5</td>
<td>4.00e-7</td>
<td>4.14e-8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Forward algorithm

\[ f_l(i + 1) = p_l(x_{i+1}) \cdot \sum_k f_k(i) \cdot a_{kl} \]

\[ P(x) = \sum_k f_k(L) \]

\[ P(x) = (1.79 + 0.414) \cdot 10^{-7} = 2.2 \cdot 10^{-7} \]

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>8.30e-2</td>
<td>1.40e-2</td>
<td>2.63e-3</td>
<td>6.08e-4</td>
<td>1.82e-4</td>
<td>3.66e-5</td>
<td>6.48e-6</td>
<td>1.09e-6</td>
<td>1.79e-7</td>
</tr>
<tr>
<td>L</td>
<td>5.00e-2</td>
<td>2.46e-2</td>
<td>1.14e-2</td>
<td>5.20e-3</td>
<td>4.71e-4</td>
<td>4.33e-5</td>
<td>4.08e-6</td>
<td>4.00e-7</td>
<td>4.14e-8</td>
</tr>
</tbody>
</table>
The Posterior decoding  
(Backward algorithm)

• One other interesting question would be
  - What is the probability that an observation \( x_i \)
    came from a state \( k \) given the observed sequence \( x \)

\[ P(\pi_i = k \mid x) \]
The Backward algorithm

\[ P(x,\pi_i = k) = P(x_1, x_2, \ldots, x_i, \pi_i = k) \cdot P(x_{i+1}, \ldots, x_L | \pi_i = k) \]

The probability of generation the sequence up to and including \( x_i \) ending in state \( k \)  
Forward algorithm!

The probability of generation the rest of the sequence starting from state \( k \)  
Backward algorithm!
The Backward algorithm

\[
P(x, \pi_i = k) = P(x_1, x_2, ..., x_i, \pi_i = k) \cdot P(x_{i+1}, ..., x_L | \pi_i = k)
\]
\[
P(x, \pi_i = k) = f_k(i) \cdot b_k(i)
\]

\[
f_k(i) = P(x_1, x_2, ..., x_i, \pi_i = k)
\]
\[
f_k(i) = p_k(x_i) \cdot \sum_l f_l(i-1) \cdot a_{lk}
\]

\[
b_k(i) = P(x_{i+1}, x_{i+2}, ..., x_L | \pi_i = k)
\]
\[
b_k(i) = \sum_l a_{kl} \cdot p_l(x_{i+1}) \cdot b_l(i+1)
\]

\[
P(\pi_i = k | x) = \frac{P(x, \pi_i = k)}{P(x)} = \frac{f_k(i) \cdot b_k(i)}{P(x)}
\]
Backward algorithm

\[ b_k(i) = \sum_{l} a_{kl} \cdot p_l(x_{i+1}) \cdot b_l(i+1) \]

- \( b_5(5) = 0.167 \cdot 0.95 \cdot 1 + 0.1 \cdot 0.05 \cdot 1 = 0.163 \)
- \( b_5(6) = 0.167 \cdot 0.1 \cdot 1 + 0.1 \cdot 0.9 \cdot 1 = 0.107 \)

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>6</th>
<th>6</th>
<th>6</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.163</td>
<td>1</td>
</tr>
<tr>
<td>L</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.107</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

- **Fair**: 1.1/6, 2.1/6, 3.1/6, 4.1/6, 5.1/6, 6.1/6
- **Loaded**: 1.1/10, 2.1/10, 3.1/10, 4.1/10, 5.1/10, 6.1/2

0.95 - 0.9 - 0.05 - 0.1 - 0.10 - 0.163 - 0.107
Backward algorithm

\[ b_k(i) = \sum_l a_{kl} \cdot p_l(x_{i+1}) \cdot b_l(i+1) \]
Backward algorithm

- Note that the sum of the first column of the backward matrix is NOT equal to the sum of the last column of the forward matrix.
- This is because the first column of the backward matrix gives the probability values of generating the sequence AFTER having generated the first observation.
- You hence cannot get the $P(x)$ value directly from the backward matrix.
Posterior decoding

• What is the posterior probability that observation $x_i$ came from state $k$ given the observed sequence $X$?

or

• What is the probability that a given amino acids is part of the trans-membrane helix given the protein sequence is $X$?
Posterior decoding

- What is the posterior probability that observation $x_i$ came from state $k$ given the observed sequence $X$.

$$P(\pi_i = k \mid x) = \frac{P(x, \pi_i = k)}{P(x)} = \frac{f_k(i) \cdot b_k(i)}{P(x)}$$
Posterior decoding

The probability is context dependent

\[
P(\pi_i = k \mid x) = \frac{P(x, \pi_i = k)}{P(x)} = \frac{f_k(i) \cdot b_k(i)}{P(x)}
\]
Training of HMM

• Supervised training
  - If each symbol is assigned to one state, all parameters can be found by simply counting number of emissions and transitions as we did for the DNA model

• Un-supervised training
  - We do not know to which state each symbol is assigned so another approach is needed
  - Find emission and transition parameters that most likely produces the observed data
  - Baum-Welsh does this
Supervised learning

- 5 matches. A, 2xC, T, G
- 5 transitions in gap region
  - C out, G out
  - A-C, C-T, T out
  - Out transition 3/5
  - Stay transition 2/5

\[
ACA---ATG \quad 0.8 \times 1 \times 0.8 \times 1 \times 0.8 \times 0.4 \times 1 \times 1 \times 0.8 \times 1 \times 0.2 = 3.3 \times 10^{-2}
\]
Un-supervised learning

312453666641456667543
5666663331234

Fair

1:e11
2:e12
3:e13
4:e14
5:e15
6:e16

Loaded

1:e21
2:e22
3:e23
4:e24
5:e25
6:e26

a11
a12
a21
a22
The probability of being in state k at time i, and state l at time i + 1, given the model and the observation sequence is

$$\varepsilon_{kl}^i = \frac{1}{P(x)} \cdot f_k(i) \cdot a_{kl} \cdot e_l(x_{i+1}) \cdot b_l(i+1)$$

The probability of being in state k at time i, given the observation sequence O is

$$\gamma_k^i = \frac{1}{P(x)} \cdot f_k(i) \cdot b_k(i)$$

Note

$$\gamma_k^i = \sum_l \varepsilon_{kl}^i$$
Now

\[ p(k) = \sum_{i=1}^{T-1} \gamma_k^i \]

is the expected number of times that state \( k \) is visited, or the expected number of transitions made from state \( k \) (given the observed sequence), and

\[ \sum_{i=1}^{T-1} \varepsilon_{kl}^i \]

is the expected number of transitions from state \( k \) to state \( l \) (given the observed sequence)
Now
\[
a_{kl} = \frac{\sum_{i=1}^{T-1} \epsilon_{kl}^i}{\sum_{i=1}^{T-1} \gamma_k^i}
\]

Estimate probability of transition between state k and l

and

\[
e_{a_k}^a = \frac{\sum_{i=1, O_i = v_a}^T \gamma_k^i}{\sum_{i=1}^T \gamma_k^i}
\]

Estimate probability emitting symbol a in state k
Baum-Welsh

Use these relations

\[ a_{kl} = \frac{\sum_{i=1}^{T-1} \varepsilon_{kl}^i}{\sum_{i=1}^{T-1} \gamma_k^i} \]

\[ e^a_k = \frac{\sum_{i=1, O_i = v_a}^{N-1} \gamma_k^i}{\sum_{i=1}^{N-1} \gamma_k^i} \]

To update \( a \), and \( e \), and iterate until convergence
HMM’s and weight matrices

• In the case of un-gapped alignments, HMM’s become simple weight matrices.
• To achieve high performance, the emission frequencies are estimated using the techniques of:
  - Sequence weighting
  - Pseudo counts
Profile HMM’s

• Alignments based on conventional scoring matrices (BLOSUM62) scores all positions in a sequence in an equal manner

• Some positions are highly conserved, some are highly variable (more than what is described in the BLOSUM matrix)

• Profile HMM’s are ideal suited to describe such position specific variations
Sequence profiles

Matching any thing but G => large negative score

Any thing can match
HMM vs. alignment

- Detailed description of core
  - Conserved/variable positions
- Price for insertions/deletions varies at different locations in sequence
- These features cannot be captured in conventional alignments
Profile HMM’s

All P/D pairs must be visited once

$L_1 - Y_2A_3V_4R_5 - I_6$
$P_1D_2P_3P_4I_4P_5D_6P_7$
Profile HMM

• Un-gapped profile HMM is just a sequence profile
Profile HMM

• Un-gapped profile HMM is just a sequence profile

\[
\begin{align*}
P1 & \rightarrow P2 \rightarrow P3 \rightarrow P4 \rightarrow P5 \rightarrow P6 \rightarrow P7 \\
A & = 0.05 \\
C & = 0.01 \\
D & = 0.08 \\
E & = 0.08 \\
F & = 0.03 \\
G & = 0.02 \\
\ldots & \\
V & = 0.08 \\
Y & = 0.01 \\
\end{align*}
\]

\[a_{lk} = 1.0\]
Example. Where is the active site?

- Sequence profiles might show you where to look!
- The active site could be around:
  - S9, G42, N74, and H195
Profile HMM

• Profile HMM (deletions and insertions)
Profile HMM (deletions and insertions)

QUERY  HAMDIRCYHSGG-PLHL-GEI-EDFNGQSCIVCPWHKYKITLATGE--GLYQSINPKDPS
Q8K2P6  HAMDIRCYHSGG-PLHL-GEI-EDFNGQSCIVCPWHKYKITLATGE--GLYQSINPKDPS
Q8TAC1  HAMDIRCYHSGG-PLHL-GDI-EDFDGRPCIVCPWHKYKITLATGE--GLYQSINPKDPS
Q07947  FAVQDTCTHGDW-ALSE-GYI-DGD----VVECTLHFGKFVCRTGK--VKAL------PA
P0A185  YATDNLCTHGSA-RMSD-GYI-EGRE----IECPLHQGRFDVCTGK--ALC------APV
P0A186  YATDNLCTHGSA-RMSD-GYI-EGRE----IECPLHQGRFDVCTGK--ALC------APV
Q51493  YATDNLCTHGAA-RMSD-GFI-EGRE----IECPLHQGRFDVCTGR--ALC------APV
A5W4F0  FAVQDTCTHGDW-ALSD-GYI-DGD----IVECTLHFGKFCVRTGK--VKAL------PA
P0C620  FAVQDTCTHGDW-ALSD-GYI-DGD----IVECTLHFGKFCVRTGK--VKAL------PA
P08086  FAVQDTCTHGDW-ALSD-GYI-DGD----IVECTLHFGKFCVRTGK--VKAL------PA
Q52440  FATQDQCTHGEW-SLSE-GGY-LDGD---VVECSLHMGKFCVRTG------------V
Q7N4V8  FAVDDRCSHGNA-SISE-GYL-ED---NATVECPLHTASFLRTGK--ALCL------PA
P37332  FATQDRCTHGDW-SLSD-GYI-EGD---VVECSLHMGKFCVRTG-------------V
A7ZPY3  YAINDRCSHGNA-SMSE-GYI-EDD---ATVECPLHAASFCILTGK--ALCL------PA
P0ABW1  YAINDRCSHGNA-SMSE-GYI-EDD---ATVECPLHAASFCILTGK--ALCL------PA
A8A346  YAINDRCSHGNA-SMSE-GYI-EDD---ATVECPLHAASFCILTGK--ALCL------PA
P0ABW0  YAINDRCSHGNA-SMSE-GYI-EDD---ATVECPLHAASFCILTGK--ALCL------PA
P0ABW2  YAINDRCSHGNA-SMSE-GYI-EDD---ATVECPLHAASFCILTGK--ALCL------PA
Q3YZ13  YAINDRCSHGNA-SMSE-GYI-EDD---ATVECPLHAASFCILTGK--ALCL------PA
Q06458  YALDNLEPGSEANVLSR-GLL-GDAGGEPIVISPLYKQRIRLDRG----------

Core  Insertion  Deletion
The HMMer program

- HMMer is an open source program suite for profile HMM for biological sequence analysis
- Used to make the Pfam database of protein families
  - http://pfam.sanger.ac.uk/
A HMMer example

- Example from the CASP8 competition
- What is the best PDB template for building a model for the sequence T0391

> T0391 rieske ferredoxin, mouse, 157 residues
SDPEISEQDEEKKKYTSVCVGREEDIRKSERMTAVVHDREVVIYHKGEYHAMDIRCYHS
GGPLHLGEIEDFNGQSCIVCPWHKYKITLATGEGLYQSINPKDPASAKPWCSKGVKQRIH
TVKVDNIGNIYVTLSKEPFKCSDYYATGEFKVIQSSS
A HMMer example

• What is the best PDB template for building a model for the sequence T0391
  • Use Blast
    - No hits
  • Use Psi-Blast
    - No hits
  • Use Hmmer
A HMMer example

• Use Hmmer
  - Make multiple alignment using Blast
  - Make model using
    • hmmbuild
  - Find PDB template using
    • hmmsearch
A HMMer example

• Make multiple alignment using Blast
  blastpgp -j 3 -e 0.001 -m 6 -i T0391.fsa -d sp -b
  10000000 -v 10000000 > T0391.fsa.blastout

• Make Stockholm format
  # STOCKHOLM 1.0
  QUERY DPEISEQDEEKKYTSCVGREEDIRKS-ERMTAVVHD-RE--V-V-IF--Y-H-KGE-Y
  Q8K2P6 DPEISEQDEEKKYTSCVGREEDIRKS-ERMTAVVHD-RE--V-V-IF--Y-H-KGE-Y
  Q8TAC1 ----SAQDPEKREYSSVCVGREDDIKKS-ERMTAVVHD-RE--V-V-IF--Y-H-KGE-Y

• Build HMM
  hmmbuild T0391.hmm T0391.fsa.blastout.sto

• Search for templates in PDB
  hmmsearch T0391.hmm pdb > T0391.out
## A HMMer example

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Description</th>
<th>Score</th>
<th>E-value</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D89.A</td>
<td>mol:aa ELECTRON TRANSPORT</td>
<td>178.6</td>
<td>2.1e-49</td>
<td>1</td>
</tr>
<tr>
<td>2E4Q.A</td>
<td>mol:aa ELECTRON TRANSPORT</td>
<td>163.7</td>
<td>6.7e-45</td>
<td>1</td>
</tr>
<tr>
<td>2E4P.B</td>
<td>mol:aa ELECTRON TRANSPORT</td>
<td>163.7</td>
<td>6.7e-45</td>
<td>1</td>
</tr>
<tr>
<td>2E4P.A</td>
<td>mol:aa ELECTRON TRANSPORT</td>
<td>163.7</td>
<td>6.7e-45</td>
<td>1</td>
</tr>
<tr>
<td>2E4Q.C</td>
<td>mol:aa ELECTRON TRANSPORT</td>
<td>163.7</td>
<td>6.7e-45</td>
<td>1</td>
</tr>
<tr>
<td>2YVJ.B</td>
<td>mol:aa OXIDOREDUCTASE/ELECTRON TRANSPORT</td>
<td>163.7</td>
<td>6.7e-45</td>
<td>1</td>
</tr>
<tr>
<td>1FQT.A</td>
<td>mol:aa OXIDOREDUCTASE</td>
<td>160.9</td>
<td>4.5e-44</td>
<td>1</td>
</tr>
<tr>
<td>1FQT.B</td>
<td>mol:aa OXIDOREDUCTASE</td>
<td>160.9</td>
<td>4.5e-44</td>
<td>1</td>
</tr>
<tr>
<td>2QPZ.A</td>
<td>mol:aa METAL BINDING PROTEIN</td>
<td>137.3</td>
<td>5.6e-37</td>
<td>1</td>
</tr>
<tr>
<td>2Q3W.A</td>
<td>mol:aa ELECTRON TRANSPORT</td>
<td>116.2</td>
<td>1.3e-30</td>
<td>1</td>
</tr>
<tr>
<td>1VM9.A</td>
<td>mol:aa ELECTRON TRANSPORT</td>
<td>116.2</td>
<td>1.3e-30</td>
<td>1</td>
</tr>
</tbody>
</table>
A HMMer example

This is the structure we are trying to predict
Validation. CE structural alignment

CE 2E4Q A 3D89 A (run on IRIX machines at CBS)


CE Algorithm, version 1.00, 1998.

Chain 1: /usr/cbs/bio/src/ce_distr/data.cbs/pdb2e4q.ent:A (Size=109)
Chain 2: /usr/cbs/bio/src/ce_distr/data.cbs/pdb3d89.ent:A (Size=157)

Alignment length = 101
Rmsd = 2.03Å
Z-Score = 5.5
Gaps = 20 (19.8%)
CPU = 1s
Sequence identities = 18.1%

Chain 1: 2 TFTKACSVDEVPPGEALQVSHDAQKVKAIFNVDGEFFATQDQCTHGEWSLSEGGYLDG-----DVVECSLHM
Chain 2: 16 TSVCVGREEDIRKSERMTAVVHDREVVIFYHKGYHAMDIRCYHSGGPLH-LGEIEDFNGQQSCIVCPWHK

Chain 1: 68 GKFCVRTGKVKS-----PPPC-----------EPLKVYPIRERGVDVLDFDSRAALH
Chain 2: 85 YKITLATGEGLYQSINPKDPSAKPKWCSKGVQRIHTVVDNGNIYVTL-SKEPF
HMM packages

• **HMMER** (http://hmmer.wustl.edu/)
  - S.R. Eddy, WashU St. Louis. Freely available.
• **SAM** (http://www.cse.ucsc.edu/research/compbio/sam.html)
  - R. Hughey, K. Karplus, A. Krogh, D. Haussler and others, UC Santa Cruz. Freely available to academia, nominal license fee for commercial users.
• **META-MEME** (http://metameme.sdsc.edu/)
• **NET-ID, HMMpro** (http://www.netid.com/html/hmmpro.html)
  - Freely available to academia, nominal license fee for commercial users.
  - Allows HMM architecture construction.

• **EasyGibbs** (http://www.cbs.dtu.dk/biotools/EasyGibbs/)
  - Webserver for Gibbs sampling of proteins sequences