Practical use of HMMs and Protein Sequence Databases
Outline

Part 1: HMMs for proteins

- Practical HMM usage: Hmmer, E/p-values and limitations
- HMMs for family alignment and for secondary structure prediction

Part 2: The Protein Universe

- Tour of the protein sequence space: Pfam, Interpro, SCOP
- Protein Evolution: Observations

References:

- Biological sequence analysis
- HMMER User guide
- Fundamentals of Molecular Evolution (Graur/Li)
- The Neutral Theory of Molecular Evolution (Kimura)
Review: HMMs (Hidden Markov Models)

- System moves through a series of “hidden” states with specified *transition probabilities*
- System “emits” observables, with different *emission probabilities* in each hidden state.

Occasionally dishonest casino

Important Algorithms:
- **Forward algorithm**: Compute probability of a sequence of observables, summing over all possible paths through hidden states
- **Viterbi Algorithm**: Compute the most likely path though hidden states, given a sequence of observables
- **Baum–Welch algorithm**: Given a set of observed sequences and a topology, estimate emission and transition probabilities (build an HMM).
There are many possible choices of “topology” for HMMs, depending on the application. We will focus on two types:

1. Secondary-Structure HMMs (SAM, OSS-HMM, TMHMM)

2. Profile HMMs (HMMER)
   (for multiple sequence alignments)
Want to model which parts of a protein sequence are transmembrane helices.

Simple Two-State Model

\[ X = \text{Everything else} \quad M = \text{TM helix} \]

Transition Probabilities

<table>
<thead>
<tr>
<th>Transition</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>M (\rightarrow) X</td>
<td>0.046</td>
</tr>
<tr>
<td>M (\rightarrow) M</td>
<td>0.954</td>
</tr>
<tr>
<td>X (\rightarrow) X</td>
<td>0.982</td>
</tr>
<tr>
<td>X (\rightarrow) M</td>
<td>0.015</td>
</tr>
</tbody>
</table>

Emission Probabilities

<table>
<thead>
<tr>
<th>Amino acids</th>
<th>TM helices</th>
<th>Other regions</th>
<th>Over-represented</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Count</td>
<td>Frequency</td>
<td>Count</td>
</tr>
<tr>
<td>I</td>
<td>1826</td>
<td>0.120</td>
<td>2187</td>
</tr>
<tr>
<td>F</td>
<td>1370</td>
<td>0.090</td>
<td>1854</td>
</tr>
<tr>
<td>L</td>
<td>2562</td>
<td>0.168</td>
<td>4156</td>
</tr>
<tr>
<td>V</td>
<td>1751</td>
<td>0.115</td>
<td>2935</td>
</tr>
<tr>
<td>M</td>
<td>616</td>
<td>0.040</td>
<td>1201</td>
</tr>
<tr>
<td>W</td>
<td>414</td>
<td>0.027</td>
<td>819</td>
</tr>
<tr>
<td>A</td>
<td>1657</td>
<td>0.109</td>
<td>3382</td>
</tr>
<tr>
<td>Y</td>
<td>615</td>
<td>0.040</td>
<td>1616</td>
</tr>
<tr>
<td>G</td>
<td>1243</td>
<td>0.082</td>
<td>3352</td>
</tr>
<tr>
<td>C</td>
<td>289</td>
<td>0.019</td>
<td>960</td>
</tr>
<tr>
<td>T</td>
<td>755</td>
<td>0.050</td>
<td>2852</td>
</tr>
<tr>
<td>S</td>
<td>806</td>
<td>0.053</td>
<td>3410</td>
</tr>
<tr>
<td>P</td>
<td>423</td>
<td>0.028</td>
<td>2640</td>
</tr>
<tr>
<td>H</td>
<td>121</td>
<td>0.008</td>
<td>1085</td>
</tr>
<tr>
<td>N</td>
<td>250</td>
<td>0.016</td>
<td>2279</td>
</tr>
<tr>
<td>Q</td>
<td>141</td>
<td>0.009</td>
<td>2054</td>
</tr>
<tr>
<td>D</td>
<td>104</td>
<td>0.007</td>
<td>2551</td>
</tr>
<tr>
<td>E</td>
<td>110</td>
<td>0.007</td>
<td>2983</td>
</tr>
<tr>
<td>K</td>
<td>78</td>
<td>0.005</td>
<td>2651</td>
</tr>
<tr>
<td>R</td>
<td>83</td>
<td>0.005</td>
<td>2933</td>
</tr>
<tr>
<td>Total</td>
<td>15214</td>
<td>1.000</td>
<td>47900</td>
</tr>
</tbody>
</table>
Simple Transmembrane Helix HMM

Want to model which parts of a protein sequence are transmembrane helices.

Simple Two-State Model

\[ X = \text{Everything else} \quad M = \text{TM helix} \]

Can improve with more complex topologies

“Real” high performance HMMs can be much more complicated.

“Correctly predicts 97-98 % of transmembrane helices.”

Used to estimate that 20-30 % of all genes are TM proteins

Posterior probabilities for a single sequence:
HMMS for Secondary Structure Prediction

- Naive idea: 3 state topology: Helix, Coil, Strand
- Real World: 36 hidden states (OSS-HMM)
- Topology not chosen by hand: An algorithm tries topologies to find the one with best match with DSSP predictions

Similar Performance to PSIPRED

Insights into secondary structure patterns?
Profile HMMs (HMMER)

HMMs used to model and align Multiple Sequence Alignments

- Topology of a profile HMM
- Building the HMM
- Using the HMM to align sequences
- Using the HMM to search for sequences in a database
- Tool: HMMER
- Practical challenges to building a profile HMM given an MSA
Profile HMMs (HMMER)

Review of Topology:

- **Delete states (gap)**
- **Insert states (emit aa, with unbiased probabilities)**
- **Match states (emit aa, with position-specific probabilities)**

Helix

```
AAAAAADDAAAAAAA       BBBBBBBBBBDDDDDDDDDDDDDDDDDDDDDDDDDD
HBA_HUMAN             -----------------VLSHAPKTNKAAWOKVGA----HAGWEYGAERLRMFLSFPPTTKYFPHF
HBB_HUMAN             -----------------VHLTPEEKSATLWKG---------NDEVGGEALGRLLVYYPWTQRFESSF
MYG_PHYCA             -----------------VLSFEGWQLVHVVWAKVEA-----DVAGHGDILDILRFLPKSHPETLEKFDRF
GLB3_CHITP            -----------------LSADQISTVQSFDKVKG-------DPVGLYAVFKAPSMAMKFTQF
GLB5_PETMA            PIVDTGSPVPLSAAEKTKIRSAWAPVYS--TyTSGVDILVIFSTSTPAQAQFEFPKFP
LGB2_LUPLU            -----------------GALTESSQALVSSWEEFNA----NIPKHTHRFFILVLEIAPKDLFS-F
GLB1_GLYDI            -----------------GLSAAQRQVIAATWKDIAGDNGAGVGKDCLKFLSAPMQAMAVFG-F
Consensus             Ls.... v a W k v . . g . L.. f . P . F F
```
Profile HMMs  (HMMER)

Example of a profile HMM with all values illustrated:

Weight of line = transition probability

```
FPHF-DLS------HGSAQ
FESFGDLSTPDAMGNPK
FDRFKHKLKTEAEMKASED
FTQFAG-KDLESIKGTAP
FPKFGLTADOLKKSAD
FS-FLK-GTSEVPQNNPE
FG-FSG------AS--DPG
```

Unaligned (insert)
What you can do with profile HMMs:

Basic (Given a Model):
- Score Sequences:
  - Compute probability that a sequence was generated by the HMM
- Search For Matching Sequences
  - Score all sequences in a database, pick only the high scoring ones
- Compute Site Posterior Probability:
  - Reflects the degree of confidence in each individual aligned residue
- Align Sequences to known model:
  - Find the most probable alignment of a unaligned sequence (viterbi)

Higher-level (No initial model):
- Build a model given an aligned MSA
  - simple counting
- Build a model given unaligned sequences
  - Baum-Welch algorithm (iterates forward/backward algorithms)
- Iterative model building/search
  - Iteratively run Baum-Welch, and collect new sequences from a database using it.
Scoring Sequences

What is the probability that a given sequence was generated by the model?

We have two types of probabilities:

- We can calculate the probability of a sequence through most likely path through hidden states (**Viterbi algorithm**)

\[
P(x, \pi^* | M)
\]

- Or, we can calculate the probability of a sequence averaging over all paths (**forward algorithm**)

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

\[\text{Observed sequence}\]

\[\text{Viterbi path}\]

\[\text{Model}\]

In practice we use the second: “To maximize power to discriminate true homologs from nonhomologs in a database search, statistical inference theory says you ought to be scoring sequences by integrating over alignment uncertainty, not just scoring the single best alignment”

We can compare this to the probability that it would be generated by a “random” model

- Null model (random aa)

\[
P(x | R) = \prod_{i} q_{x_i}
\]

mean residue frequencies in Swiss-Prot 50.8
Scoring Sequences: Log Odds Ratio

The probability of a sequence averaging over all paths (forward algorithm)

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

Probability of sequence in Null model (random amino acids)

\[ P(x|\text{R}) = \prod_{i} q_{x_{i}} \]

The “Log odds ratio”
(“sequence bit score”)

\[ S(x) = \log \frac{P(x)}{P(x|\text{R})} \]

- Standard way of testing model discriminatory power
- Values >> 1 imply the sequence is more consistent with the HMM
- We take the log to avoid numerical difficulties (numbers can be very small)
- When the log is base 2, the value has units called “bits”.
- Log-odds ratio is also called the “sequence bit score”
- Can be efficiently computed for HMMs using a modified Forward algorithm
Scoring Sequences: E-Values

The “Log odds ratio”

\[ S(x) = \log \frac{P(x)}{P(x|R)} \]

Problem: The log-odds ratio gives us a score for each individual sequence, but if you are testing many, many non-matching sequences, some will still have high scores by chance.

**E-values:** Expected number of sequences having this bit score or more, when looking in a dataset of size \( N \).

Theory: Can work out that the “null” distribution of bit-scores is exponential

\[ P(S > t) \propto \mu e^{-\lambda(t-\tau)} \]

This is a way to estimate if a high score appeared by chance due to having a large dataset.

\[ \lambda = \log 2 + \frac{1.44}{hL} \quad \text{, necessary to calibrate } \mu/\tau \]
Given an HMM, and a sequence database of size N to search through, we can calculate:

- A **log-odds ratio (bit score)** for each sequence.
  
  >1 means it’s more likely to be generated by the HMM than the null model

- An **E-Value**: Probability a non-match sequence has this score by chance.

  E-value < 1 means that on average we don’t expect this strong a score by chance given our dataset size, so it is likely a homolog. Commonly E < 0.01 is used as a significance cutoff

By setting a threshold on the E-Value, we can filter the database to pick out the homologous sequences.
Posterior Probabilities

- Per-site **posterior probability annotation**.
  Probability that emitted observable \( i \) came from state \( k \), averaging over all possible paths.

Computable from forward/backward equations.

\[
P(\pi_i = k \mid M, x) = \frac{f_k(i)b_k(i)}{P(x)}
\]

Gives an estimate of how “well aligned” a position is.
A different way to align sequences?

Naive way: **Viterbi Algorithm:**
Computes the single most likely path through the HMM.
(maximizes sum of log probabilities)

Problem: The single most likely path might skip a node which almost all other likely paths go through. i.e., it can go through “unlikely” nodes. It is an extremal path rather than an average.

Better way: **MAC Algorithm:** (Maximum ACcuracy)
Computes the path with the greatest total posterior probability through the HMM
(maximizes sum of posterior probabilities)

i.e., computes the path going through the most commonly visited nodes by all likely paths, and so is a better representation of the “average path”.

- Easy implementation: Very similar to the Viterbi algorithm, except we substitute the posterior probabilities instead of the log probabilities, plus minor tweaks.

- Gives “more accurate” alignments than Viterbi. MAC is used by HMMER
What you can do with profile HMMs:

Basic (Given a Model):
- Score Sequences:
  - Compute probability that a sequence was generated by the HMM
- Search For Matching Sequences
  - Score all sequences in a database, pick only the high scoring ones
- Compute Site Posterior Probability:
  - Reflects the degree of confidence in each individual aligned residue
- Align Sequences to known model:
  - Find the most probable alignment of a unaligned sequence (viterbi)

Higher-level (No initial model):
- Build a model given an aligned MSA
  - simple counting
- Build a model given unaligned sequences
  - Baum-Welch algorithm (iterates forward/backward algorithms)
- Iterative model building/search
  - Iteratively run Baum-Welch, and collect new sequences from a database using it.
HMM building Algorithms

Building an HMM given an aligned MSA

*Idea:* Very simply, we count up what we see in the MSA. i.e., count up the number of times each amino acid at each position, and normalize, to get the emission probabilities, and similarly for transition probabilities. Pseudocounts and weights may be used too (discussed later).

Building an HMM given an unaligned MSA (The Baum-Welch Algorithm)

*Idea:* Given a trial HMM, use forward/backward algorithms to average over all possible alignments of all sequences in the MSA. Add up the observed emission and transitions seen in the data with these probabilities.

Building an HMM given a seed MSA and a database (JACKHMMER, HHBlits)

*Idea:* Given a seed aligned MSA, construct an initial HMM (see above). Use this HMM to search the database for more homologous sequences, to get a bigger, more diverse MSA. Construct a new HMM with this MSA. Repeat.
HMMER

- Free Software for using/building profile HMMs
- HMMER3 is highly optimized. Fast!

The programs in HMMER

**Build models and align sequences (DNA or protein)**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hmmbuild</td>
<td>Build a profile HMM from an input multiple alignment.</td>
</tr>
<tr>
<td>hmmalign</td>
<td>Make a multiple alignment of many sequences to a common profile HMM.</td>
</tr>
</tbody>
</table>

**Search protein queries against protein database**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phmmer</td>
<td>Search a single protein sequence against a protein sequence database. (BLASTP-like)</td>
</tr>
<tr>
<td>jackhmm</td>
<td>Iteratively search a protein sequence against a protein sequence database. (PSIBLAST-like)</td>
</tr>
<tr>
<td>hmmsearch</td>
<td>Search a protein profile HMM against a protein sequence database.</td>
</tr>
<tr>
<td>hmmscar</td>
<td>Search a protein sequence against a protein profile HMM database.</td>
</tr>
<tr>
<td>hmmmpgmd</td>
<td>Search daemon used for hmmmer.org website.</td>
</tr>
</tbody>
</table>

**Search DNA queries against DNA database**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nhmmer</td>
<td>Search a DNA sequence, alignment, or profile HMM against a DNA sequence database. (BLASTN-like)</td>
</tr>
<tr>
<td>nhmmscan</td>
<td>Search a DNA sequence against a DNA profile HMM database.</td>
</tr>
</tbody>
</table>

**Other utilities**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alimask</td>
<td>Modify alignment file to mask column ranges.</td>
</tr>
<tr>
<td>hmmconvert</td>
<td>Convert profile formats to/from HMMER3 format.</td>
</tr>
<tr>
<td>hmmemit</td>
<td>Generate (sample) sequences from a profile HMM.</td>
</tr>
<tr>
<td>hmmfetch</td>
<td>Get a profile HMM by name or accession from an HMM database.</td>
</tr>
<tr>
<td>hmmmpress</td>
<td>Format an HMM database into a binary format for hmmscan.</td>
</tr>
<tr>
<td>hmmstat</td>
<td>Show summary statistics for each profile in an HMM database.</td>
</tr>
</tbody>
</table>
HMMER: Example Run

Example: Searching a sequence database

```
> hmmsearch globins4.hmm uniprot_sprot.fasta > globins4.out
```

```
# hmmsearch :: search profile(s) against a sequence database
# HMMER 3.1 (February 2013); http://hmmer.org/
# Copyright (C) 2011 Howard Hughes Medical Institute.
# Freely distributed under the GNU General Public License (GPLv3).
#---------------------------------------------------------------
# query HMM file:  globins4.hmm
# target sequence database:  uniprot_sprot.fasta
# per-seq hits tabular output:  globins4.tbl
# per-dom hits tabular output:  globins4.domtbl
# number of worker threads:  2
#---------------------------------------------------------------

Query:  globins4  [M=149]
Scores for complete sequences (score includes all domains):

<table>
<thead>
<tr>
<th>E-value</th>
<th>score</th>
<th>bias</th>
<th>E-value</th>
<th>score</th>
<th>bias</th>
<th>#dom-</th>
<th>exp</th>
<th>N</th>
<th>Sequence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.5e-65</td>
<td>222.7</td>
<td>3.2</td>
<td>7.2e-65</td>
<td>222.6</td>
<td>3.2</td>
<td>1.0</td>
<td>1</td>
<td>sp</td>
<td>P02185</td>
<td>MYG_PHYMC</td>
</tr>
<tr>
<td>3.3e-63</td>
<td>217.2</td>
<td>0.1</td>
<td>3.7e-63</td>
<td>217.0</td>
<td>0.1</td>
<td>1.0</td>
<td>1</td>
<td>sp</td>
<td>P02024</td>
<td>HBB_GORGO</td>
</tr>
<tr>
<td>4.9e-63</td>
<td>216.6</td>
<td>0.0</td>
<td>5.4e-63</td>
<td>216.5</td>
<td>0.0</td>
<td>1.0</td>
<td>1</td>
<td>sp</td>
<td>P68871</td>
<td>HBB_HUMAN</td>
</tr>
<tr>
<td>4.9e-63</td>
<td>216.6</td>
<td>0.0</td>
<td>5.4e-63</td>
<td>216.5</td>
<td>0.0</td>
<td>1.0</td>
<td>1</td>
<td>sp</td>
<td>P68872</td>
<td>HBB_PANPA</td>
</tr>
<tr>
<td>4.9e-63</td>
<td>216.6</td>
<td>0.0</td>
<td>5.4e-63</td>
<td>216.5</td>
<td>0.0</td>
<td>1.0</td>
<td>1</td>
<td>sp</td>
<td>P68873</td>
<td>HBB_PANTR</td>
</tr>
<tr>
<td>7e-63</td>
<td>216.1</td>
<td>3.0</td>
<td>7.7e-63</td>
<td>216.0</td>
<td>3.0</td>
<td>1.0</td>
<td>1</td>
<td>sp</td>
<td>P02177</td>
<td>MYG_ESCGI</td>
</tr>
</tbody>
</table>
```
HMMER: Example Run

Example: Scoring and Aligning a sequence

HMM reference lines:
1. Annotated Secondary structure
2. Consensus sequence
3. Matches

Aligned Sequence

Posterior Probability
Profile HMMs: Practical issues

1) Local vs Global Alignments
2) Choosing Match states
3) Phylogenetic Weighting
4) Pseudocounts
5) Filtering Sequences
1. Local vs Global Alignments

**HMM for global alignment**

Try to match all residues in the query sequence to the model

*Template Match*  
WIGDEVAVKAARHDDEDIS  
WHGD-VAVKILKVVDPTPE

**HMM for local alignment**

Allows unmatched states at beginning and end, so can find a match within a longer sequence, or partial sequence.

*Template Match*  
EVAVKAAARHDDEDISvykig  
WHGD-VAVKILKVVDPTPE
2. Choosing Match States

When building a new HMM given a seed MSA, you have to make a decision: Which columns will count as match states, and which as insert states? (model construction)

- **manual construction**: The user marks alignment columns by hand.
- **heuristic construction**: A rule is used to decide whether a column should be marked. For instance, a column might be marked when the proportion of gap symbols in it is below a certain threshold.
- **MAP construction**, A maximum a posteriori choice is determined by dynamic programming

![Multiple alignment and Profile-HMM architecture](image)
2. Choosing Match States

Options Controlling Profile Construction

These options control how consensus columns are defined in an alignment.

--fast Define consensus columns as those that have a fraction $\geq \text{symfrac}$ of residues as opposed to gaps. (See below for the --symfrac option.) This is the default.

--hand Define consensus columns in next profile using reference annotation to the multiple alignment. This allows you to define any consensus columns you like.

--symfrac $<x>$ Define the residue fraction threshold necessary to define a consensus column when using the --fast option. The default is 0.5. The symbol fraction in each column is calculated after taking relative sequence weighting into account, and ignoring gap characters corresponding to ends of sequence fragments (as opposed to internal insertions/deletions). Setting this to 0.0 means that every alignment column will be assigned as consensus, which may be useful in some cases. Setting it to 1.0 means that only columns that include 0 gaps (internal insertions/deletions) will be assigned as consensus.

--fragthresh $<x>$ We only want to count terminal gaps as deletions if the aligned sequence is known to be full-length, not if it is a fragment (for instance, because only part of it was sequenced). HMMER uses a simple rule to infer fragments: if the range of a sequence in the alignment (the number of alignment columns between the first and last positions of the sequence) is less than or equal to a fraction $<x>$ times the alignment length in columns, then the sequence is handled as a fragment. The default is 0.5. Setting --fragthresh 0 will define no (nonempty) sequence as a fragment; you might want to do this if you know you've got a carefully curated alignment of full-length sequences. Setting --fragthresh 1 will define all sequences as fragments; you might want to do this if you know your alignment is entirely composed of fragments, such as translated short reads in metagenomic shotgun data.
3. Phylogenetic Weighting

"God has an inordinate fondness for beetles." - J.B.S. Haldane

If a closely related group of species appears often in the dataset, this introduces a bias in our estimates of emission probabilities. Similar to overcounting. Will hurt us for distant homology detection.

Beetles with similar sequences
3. Phylogenetic Weighting

Many different strategies to “unweight” the dataset. One way (“Henikoff scheme”):

\[ k_{ia} \quad \text{Number of times “a” is seen at position i} \]

\[ m_i \quad \text{Number of different residues ever seen at position i} \]

\[ 1 / (m_i k_{ix_i}^k) \quad \text{Weight factor needed to get “equal” emission probabilities at position i} \]

\[ w_k = \sum_i \frac{1}{m_i k_{ix_i}^k} \quad \text{Average weight factor over all positions} \]

Options Controlling Relative Weights

HMMER uses an ad hoc sequence weighting algorithm to downweight closely related sequences and up-weight distantly related ones. This has the effect of making models less biased by uneven phylogenetic representation. For example, two identical sequences would typically each receive half the weight that one sequence would. These options control which algorithm gets used.

--wpb  Use the Henikoff position-based sequence weighting scheme [Henikoff and Henikoff, J. Mol. Biol. 243:574, 1994]. This is the default.


--wblosum  Use the same clustering scheme that was used to weight data in calculating BLOSUM substitution matrices [Henikoff and Henikoff, Proc. Natl. Acad. Sci 89:10915, 1992]. Sequences are single-linkage clustered at an identity threshold (default 0.62; see --wid) and within each cluster of c sequences, each sequence gets relative weight 1/c.

--wnone  No relative weights. All sequences are assigned uniform weight.

--wid <xx>  Sets the identity threshold used by single-linkage clustering when using --wblosum. Invalid with any other weighting scheme. Default is 0.62.
4. Pseudocounts

We build an HMM given a limited seed alignment. At some positions there may be residues that we didn’t see in our dataset, but exist some fraction of the time in nature, particularly when we have few sequences.

Without any corrections, our HMM would predict that these other residues could never be output. We correct for this by adding pseudocounts:

Different ways of computing emission probabilities:

\[ e_{M_j}(a) = \frac{c_{ja}}{\sum_{a'} c_{ja'}} \]

No correction

\[ e_{M_j}(a) = \frac{c_{ja} + Aq_a}{\sum_{a'} c_{ja'} + A} \]

Simple pseudocount (A=20 works well)

\[ e_{M_j}(a) = \sum_k P(k|c_j) \frac{c_{ja} + \alpha_a^k}{\sum_{a'} (c_{ja'} + \alpha_a^k)} \]

Dirichlet Pseudocount (assumes there are different “types” of positions which have different biases)

Options Controlling Priors

By default, weighted counts are converted to mean posterior probability parameter estimates using mixture Dirichlet priors. Default mixture Dirichlet prior parameters for protein models and for nucleic acid (RNA and DNA) models are built in. The following options allow you to override the default priors.

--pnone Don’t use any priors. Probability parameters will simply be the observed frequencies, after relative sequence weighting.

--plaplace Use a Laplace +1 prior in place of the default mixture Dirichlet prior.
5. Filtering

When searching a database, there are choices about which sequences to “keep”: E values, bit scores, etc.

Options Controlling Reporting Thresholds

Reporting thresholds control which hits are reported in output files (the main output, --tblout, and --domtblout). Sequence hits and domain hits are ranked by statistical significance (E-value) and output is generated in two sections called per-target and per-domain output. In per-target output, by default, all sequence hits with an E-value $\leq 10$ are reported. In the per-domain output, for each target that has passed per-target reporting thresholds, all domains satisfying per-domain reporting thresholds are reported. By default, these are domains with conditional E-values of $\leq 10$. The following options allow you to change the default E-value reporting thresholds, or to use bit score thresholds instead.

-\texttt{E} $\langle x \rangle$  In the per-target output, report target sequences with an E-value of $\leq \langle x \rangle$. The default is $10.0$, meaning that on average, about $10$ false positives will be reported per query, so you can see the top of the noise and decide for yourself if it’s really noise.

-\texttt{T} $\langle x \rangle$  Instead of thresholding per-profile output on E-value, instead report target sequences with a bit score of $\geq \langle x \rangle$.

-\texttt{domE} $\langle x \rangle$  In the per-domain output, for target sequences that have already satisfied the per-profile reporting threshold, report individual domains with a conditional E-value of $\leq \langle x \rangle$. The default is $10.0$. A conditional E-value means the expected number of additional false positive domains in the smaller search space of those comparisons that already satisfied the per-target reporting threshold (and thus must have at least one homologous domain already).

-\texttt{domT} $\langle x \rangle$  Instead of thresholding per-domain output on E-value, instead report domains with a bit score of $\geq \langle x \rangle$.

Options for Inclusion Thresholds

Inclusion thresholds are stricter than reporting thresholds. Inclusion thresholds control which hits are considered to be reliable enough to be included in an output alignment or a subsequent search round, or marked as significant (\texttt{"!"}) as opposed to questionable (\texttt{"?"}) in domain output.

-\texttt{incE} $\langle x \rangle$  Use an E-value of $\leq \langle x \rangle$ as the per-target inclusion threshold. The default is $0.01$, meaning that on average, about $1$ false positive would be expected in every $100$ searches with different query sequences.

-\texttt{incT} $\langle x \rangle$  Instead of using E-values for setting the inclusion threshold, instead use a bit score of $\geq \langle x \rangle$ as the per-target inclusion threshold. By default this option is unset.

-\texttt{incdomE} $\langle x \rangle$  Use a conditional E-value of $\leq \langle x \rangle$ as the per-domain inclusion threshold, in targets that have already satisfied the overall per-target inclusion threshold. The default is $0.01$.

-\texttt{incdomT} $\langle x \rangle$  Instead of using E-values, use a bit score of $\geq \langle x \rangle$ as the per-domain inclusion threshold.
Aside: Protein alignments are often better than DNA alignments

Illustration using “dot matrix” method for manual alignment: Put both sequences on X and Y axes, and put a dot anywhere they are the same.

Example of Dot matrix method

Dot Matrix for a nucleotide sequence

The greater number of states in protein sequences (20 vs 4) makes alignment easier
Part 2

The Protein Universe

(What you can expect to find in the Pfam Database)
Pfam

- Pfam is a collection of HMMs and MSAs of many protein families. Uses HMMs to detect homologous sequences
- Underlying sequence database (Pfamseq) is based on UniProt
- Intimately tied to HMMER
- http://pfam.xfam.org/ (European Bioinformatics Institute)

- 16,306 curated protein families (folds)
- 23 million protein sequences
- 7.6 billion residues

collected from all branches of life

Tree of life, designed to reflect the amount of diversity among branches
Uniprot database sequence statistics

(23 million Pfam protein sequences. They come from Uniprot)
Each Pfam family consists of:

- A curated **seed alignment** containing a small set of representative members of the family
- **Profile HMMs** built iteratively from the seed alignment
- an automatically generated **full alignment**, which contains all detected sequences belonging to the family

Pfam entries are classified in one of six ways:

- **Family**: A collection of related protein regions
- **Domain**: A structural unit
- **Repeat**: A short unit which is unstable in isolation but forms a stable structure when multiple copies are present
- **Motif**: A short unit found outside globular domains
- **Coiled-Coil**: Regions that predominantly contain coiled-coil motifs, regions that typically contain alpha-helices that are coiled together in bundles of 2-7.
- **Disordered**: Regions that are conserved, yet are either shown or predicted to contain bias sequence composition and/or are intrinsically disordered (non-globular).

(From Pfam website)
All fibronectin sequences in the alignment have the same “beta sandwich” fold.

~65,000 different sequences

~100 residues long

How similar do you think the sequences are (% identical aa)?
Fibronectin Type III domain MSA

Average sequence identity: 19%
Fibronectin has quite typical diversity
The Protein Universe

- Typically ~30-40% sequence identity per fold, as low as 8-9% (equal to random sequences)

The Protein sequence $\rightarrow$ structure mapping is highly degenerate

- Causes difficulty in the “twilight zone” of sequence similarity: Below 20% identity, HMMs have difficulty predicting whether sequences belong to a family.

*How much of protein sequence space has been explored by life on Earth?* Dryden et al. J Royal Society Interface (2008)
*Twilight zone of protein sequence alignments.* Rost, Protein Eng Des Sel (1999)
The Protein Universe

- Typically ~30-40% sequence identity per fold, as low as 8-9% (equal to random sequences)
- The number of sequences which fold into the same shape (fold) is enormous, but is minuscule fraction of all possible sequences

Space of all possible sequences is \( \sim 20^{400} = 10^{520} \) 20 a.a.

Space of all sequences sharing a common folded structure may be \( \sim 7^{300} = 10^{200} \) (rough intuition/guess)

**Question:** Evolution has a huge sequence space to work with, in millions of species over billions of years. How many different shapes (folds) has it generated?

*How much of protein sequence space has been explored by life on Earth?* Dryden et al. J Royal Society Interface (2008)
The Protein Universe

- Typically ~30-40% sequence identity per fold, as low as 8-9% (equal to random sequences)
- The number of sequences which fold into the same shape (fold) is enormous, but is minuscule fraction of all possible sequences
- There are 1000 to 10,000 existing protein folds across all life

Pfam: 16,384 families
SCOP: 3,464 families, 1,086 folds

(compare to 20,000 protein-coding genes in the human genome)

Protein Families and their Evolution – A Structural perspective. Annual Review of Biochemistry 2005
Expanding protein universe and its origin from the Biological Big Bang PNAS 2002
Evolutionary Origins of Protein Families

4 genomes with multiple variants of the same protein fold:
- Human
- Dog
- Chicken
- Carp

New variants/copies are generated by:
- Gene duplication (paralog)
- Speciation (ortholog)

Tree of 512 kinases in the human genome (paralogs)
Tree of c-SRC kinase across species (orthologs)
Question: What % identity do you expect for:

- Hemoglobin-α (human) vs Hemoglobin-β (human)?
- Hemoglobin-α (human) vs Hemoglobin-α (chimp)?
Question: What % identity do you expect for:

- Hemoglobin-α (human) vs Hemoglobin-β (human)? 40%
- Hemoglobin-α (human) vs Hemoglobin-α (chimp)? 100%

α vs β: 450–500 mya (jawed fish)
human vs chimp: 4-10 mya
Some Other Sequences Patterns to Be Aware of

- **Partial Gene Duplication**
  - Domain duplication
  - Invariant repeats (dose repetition)
  - Exon shuffling, pseudoxonization

- **Tandem gene duplication**

- **Pseudogenes**
  - Unprocessed vs Processed

---

**Repeat Elements**

**Globin Gene duplicates and Pseudogenes**

---

Figure 6.11  The chromosomal arrangement of the three gene families belonging to the globin superfamily of genes in humans: the α-globin family on chromosome 16, the β-globin family on chromosome 11, and myoglobin on chromosome 22. Solid black boxes denote functional genes; empty boxes denote pseudogenes.
How is sequence diversity generated?

**Observations**

More closely related species have fewer differences per (orthologous) protein.

% difference in Hemoglobin between related species

*The Neutral Theory of Molecular Evolution. Kimura 1983*
How is sequence diversity generated?

**Observations**

Substitutions occur at a constant rate

"Rate constant hypothesis" or "Molecular Clock"

(Zuckerland & Pauling 1965)

Technical detail (correction for repeated substitutions)

\[ p = 1 - e^{-K} \]

- \( p = \) percent final sequence difference
- \( K = \) number of past substitution events
How is sequence diversity generated?

Observations

• Proteins appear to accumulate substitutions at a constant rate, usually about 1 substitution per amino-acid site per billion years
How is sequence diversity generated?

**Observations**

- Proteins appear to accumulate substitutions at a constant rate, usually about 1 substitution per amino-acid site per billion years.
- Different proteins have different evolutionary rates.

### Table 4.1. Evolutionary rates in terms of amino acid substitutions. These rates are based mostly on data from mammalian order. They are expressed per amino acid site per year taking $10^{-9}$ as the unit (‘pauling’)

<table>
<thead>
<tr>
<th>Proteins</th>
<th>$k_{aa} \times 10^9/yr$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fibrinopeptides</td>
<td>8.3</td>
</tr>
<tr>
<td>Pancreatic ribonuclease</td>
<td>2.1</td>
</tr>
<tr>
<td>Lysozyme</td>
<td>2.0</td>
</tr>
<tr>
<td>Hemoglobin $\alpha$</td>
<td>1.2</td>
</tr>
<tr>
<td>Myoglobin</td>
<td>0.89</td>
</tr>
<tr>
<td>Insulin</td>
<td>0.44</td>
</tr>
<tr>
<td>Cytochrome c</td>
<td>0.3</td>
</tr>
<tr>
<td>Histone H4</td>
<td>0.01</td>
</tr>
</tbody>
</table>
How is sequence diversity generated?

Observations

- Proteins appear to accumulate substitutions at a constant rate, usually about 1 substitution per amino-acid site per billion years.
- Different proteins have different evolutionary rates.
- Different parts of a protein have different evolutionary rates.

Why is there variation in evolutionary rate?

Table 7.1. Comparison of evolutionary rates between the surface and the heme pocket in the α and β hemoglobin chains. The listed values represent the rates of amino acid substitution in units of $10^{-9}$/amino acid/year ('paulings').

<table>
<thead>
<tr>
<th>Region</th>
<th>Hemoglobin α</th>
<th>Hemoglobin β</th>
</tr>
</thead>
<tbody>
<tr>
<td>Surface</td>
<td>1.35</td>
<td>2.73</td>
</tr>
<tr>
<td>Heme pocket</td>
<td>0.165</td>
<td>0.236</td>
</tr>
</tbody>
</table>

*Causes of evolutionary rate variation among protein sites. Nature Reviews Genetics (2016)*
Summary

- Many possible sequences lead to the same fold
- Proteins in the same family/fold accumulate substitutions at a constant rate over time

Next time: How do proteins evolve? Modeling the Protein Evolutionary Process