Using Hidden Markov Models to Recognize Protein Folds

Kevin Karplus

University of California, Santa Cruz

Supported in part by NSF grant DBI-9808007, DOE grant DE-FG03-99ER62849, and NSF grant EIA-9905322

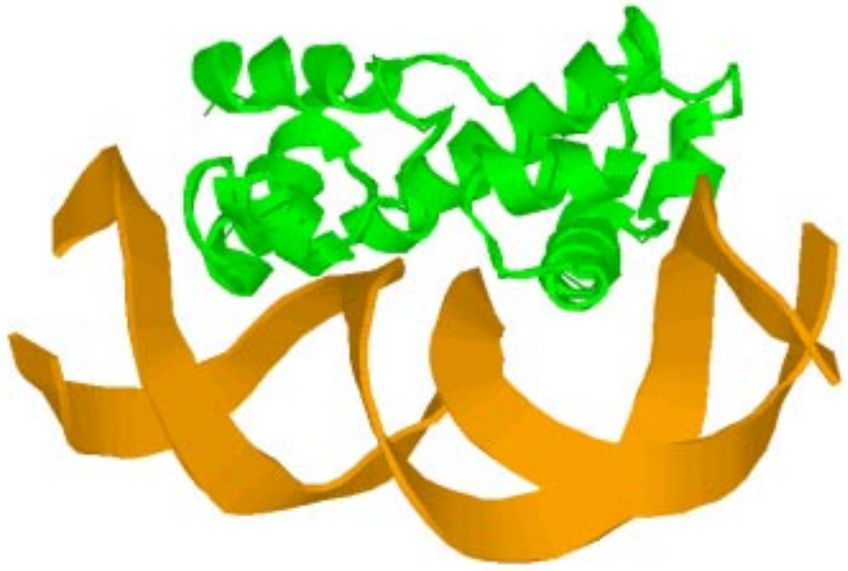
Outline of Talk

- Fold-recognition
- Scoring (Bayesian statistical modeling)
- SAM-T99 and SAM-T2K methods
- Multi-track HMMs and secondary structure
- Reverse-sequence null model
- Results

Folding Problem

The *Folding Problem*:
Given a protein expressed as a string A over the alphabet of 20 amino acids
($A \in \{a, c, d, e, f, g, h, i, k, l, m, n, p, q, r, s, t, v, w, y\}^*$),
figure out how it folds up in 3-space.

```
MTMSRRNTDA ITIHSILDWI EDNLESPLSL EKVSERSGYS KWHLQRMFKK
ETGHSLGQYI RSRKMTEIAQ KLKESNEPIL YLAERYGFES QQTLTRTFKN
YFDVPPHKYR MTNMQGESRF LHPLNHYNS
```



The *Fold-recognition Problem*:
Given a protein expressed as a string A over the alphabet of amino acids (the *target* sequence)
and a library of proteins with known 3-D structures (the *template* library),
figure out which template(s) A matches best, and align the target to the template.

- The backbone for the target sequence is predicted to be very similar to the backbone of the chosen template.
- A quality measure is needed to decide when the best-matching template is still not a good match.

The *Homology Problem*:
Given a protein expressed as a string A over the alphabet of amino acids (the *target* sequence),
and a library of protein *sequences*,
figure out which sequences A is similar to and align them to A.

- This problem is fairly easy for recently diverged, very similar sequences, but difficult for more remote relationships.
- No structure information is used, just sequence information.
- Technically, “homology” means that the sequences evolved from the same ancestral sequence—but this is almost always inferred from similarity of sequence, structure, or function, and not directly known.

- A *model* M is a computable function that assigns a probability Prob $(A \mid M)$ to each string A.
- When given a string A, we want to know how likely the model is. That is, we want to compute something like Prob $(M \mid A)$.
- Bayes Rule:

$$\text{Prob}\,(M \mid A) = \text{Prob}\,(A \mid M)\,\frac{\text{Prob}(M)}{\text{Prob}(A)}\ .$$

- Problem: Prob(A) and Prob(M) are inherently unknowable.

- Standard solution: ask how much more likely M is than some *null hypothesis* (represented by a *null model*).

$$\frac{\text{Prob}\,(M \mid A)}{\text{Prob}\,(N \mid A)} = \frac{\text{Prob}\,(A \mid M)}{\text{Prob}\,(A \mid N)} \frac{\text{Prob}(M)}{\text{Prob}(N)} .$$

- $\frac{\text{Prob}(M)}{\text{Prob}(N)}$ is the *prior odds ratio*, and represents our belief in the likelihood of the model before seeing any data.
- $\frac{\text{Prob}\left(M \middle| A\right)}{\text{Prob}\left(N \middle| A\right)}$ is the *posterior odds ratio*, and represents our belief in the likelihood of the model after seeing the data.
- We can generalize to a forced choice among many models ($M_1, \ldots, M_n$)

$$\frac{\text{Prob}\,(M_i \mid A)}{\Sigma_j\,\text{Prob}\,(M_j \mid A)} = \frac{\text{Prob}\,(A \mid M_i)\,\text{Prob}(M_i)}{\Sigma_j\,\text{Prob}\,(A \mid M_j)\,\text{Prob}(M_j)} .$$

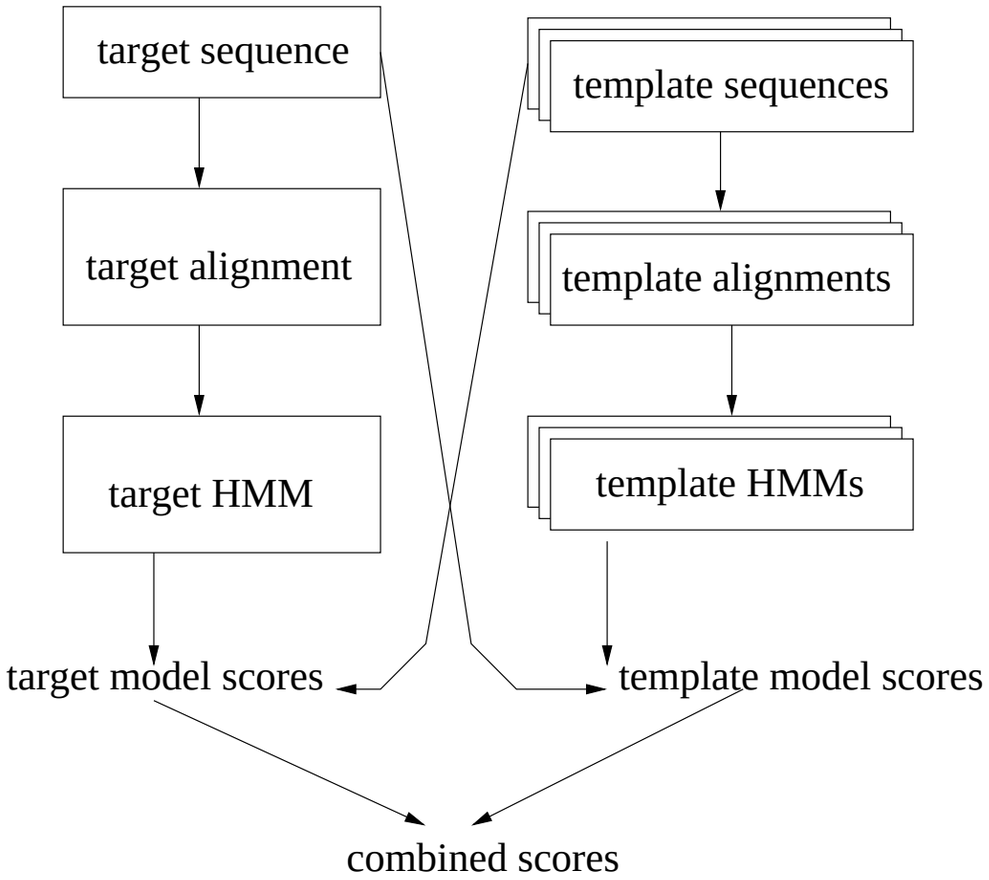
  The $\text{Prob}(M_j)$ values can be scaled arbitrarily without affecting the ratio.

- Null model is a zero-order Markov model, that is, each letter is treated as being independently drawn from the same distribution.
- $$\text{Prob}\,(A \mid N, \text{len}\,(A)) = \prod_{i=1}^{\text{len}(A)} \text{Prob}(A_i)\ .$$
- $$\text{Prob}\,(A \mid N) = \text{Prob}(\text{string of length len}\,(A)) \prod_{i=1}^{\text{len}(A)} \text{Prob}(A_i)\ .$$
- The length modeling is often omitted, but one must be careful then to normalize the probabilities correctly.

- Find probable homologs of target sequence and make multiple alignment.
- Make secondary structure probability predictions based on multiple alignment.
- Build an HMM based on the multiple alignment and predicted 2ry structure (or just on multiple alignment).
- Score sequences and secondary structure sequences for all proteins that have known structure.
- Select the best-scoring sequence(s) to use as templates.
- If the modeling method is well-chosen, the alignment of the target and template is available as a by-product of the scoring.

- Build a model for each protein in the template library, based on the template sequence (and any homologs you can find). The template library is selected as a subset of the PDB database of publicly released solved structures.
- For the fold-recognition problem, structure information can be used in building these models (though we currently don't).
- Score target sequence with all models in the library.
- Select the best-scoring model(s) to use as templates.
- Again, the alignment of the target and template may be available as a by-product of the scoring.

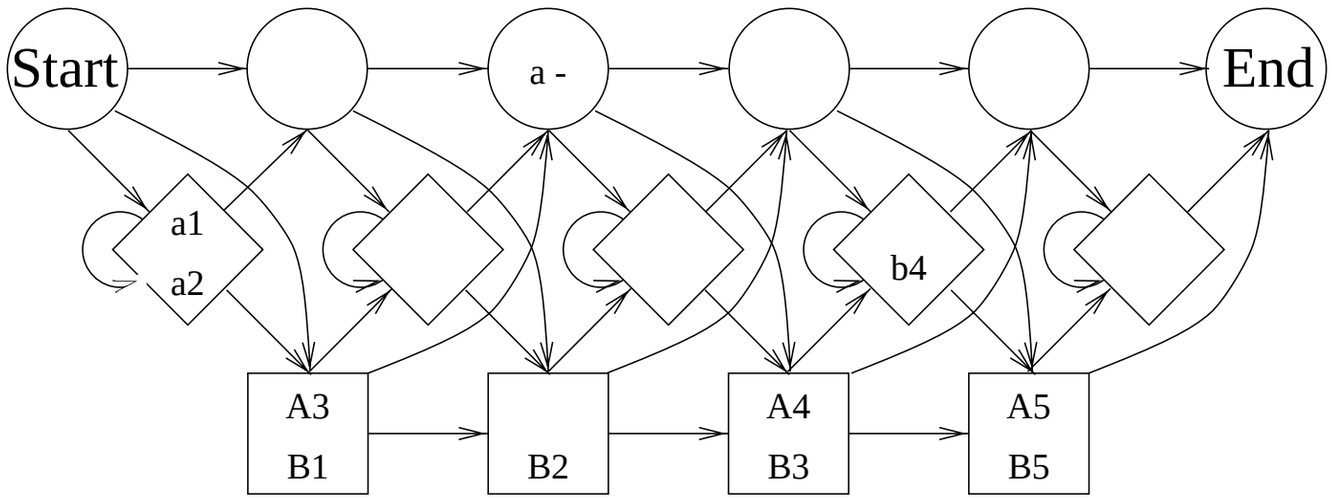
Combined SAM-Txx method



- Choose (somehow) the alignment based on the target model or the alignment based on the template model.
- This method for fold-recognition is available (with only amino-acid target HMMs, not 2-track target HMMs) on `http://www.cse.ucsc.edu/research/compbio/hmm-apps/`.
- The library currently has 5473 templates.

- A *hidden Markov Model (*HMM*)* is a finite-state machine with a probability for emitting each letter in each state, and with probabilities for making each transition between states.
- Probabilities of letters sum to one for each state.
- Probabilities of transitions out of each state sum to one for that state.
- We also include *null states* that emit no letters, but have transition probabilities on their out-edges.

Profile Hidden Markov Model




```
a1 a2 A3     -     A4  .  A5
 .  .  B1    B2    B3 b4  B5
```

- Circles are null states.
- Squares are *match states*, each of which is paired with a null *delete state*. We call the match-delete pair a *fat state*.
- Each fat state is visited exactly once on every path from Start to End.
- Diamonds are *insert states*, and are used to represent possible extra amino acids that are not found in most of the sequences in the family being modeled.

How is HMM *built?*

Overview of method for building a target HMM, given a single sequence (or a seed alignment):

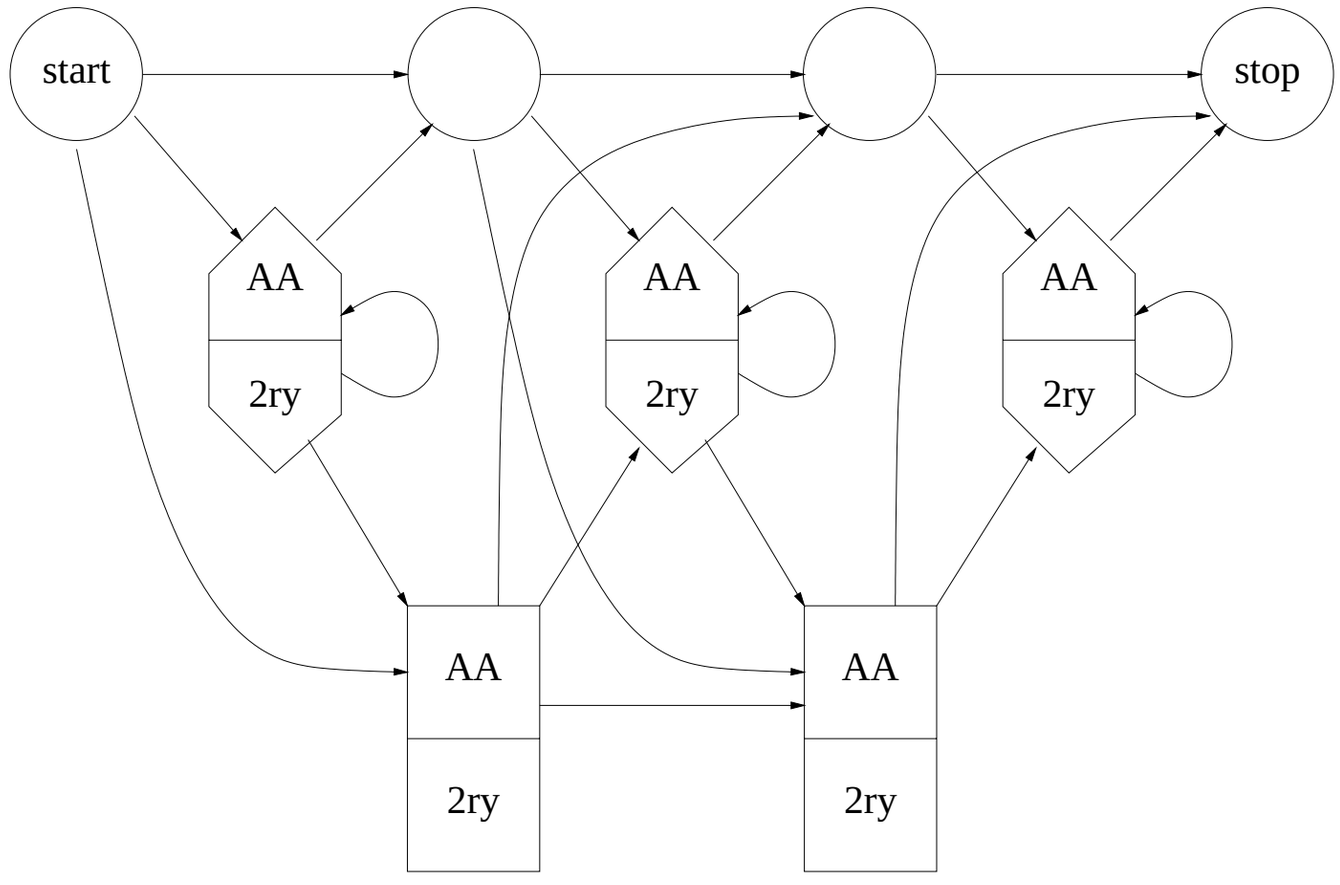
1. Construct a profile HMM with one fat state for each letter of sequence (or column of multiple alignment).
2. Find sequences in a large database of protein sequences that score well with M. This is the *training set*.
3. Retrain M (using forward-backward algorithm) to re-estimate all probabilites, based on the training set.
4. Make a multiple alignment (using Viterbi algorithm) of all sequences in the training set. The multiple alignment has one alignment column for each fat state of the HMM.
5. Repeat from step 1, with thresholds in step 2 loosened.

- Do weighting of sequences to reduce the effect of biased sampling in the database.
- Compute Prob $(a \mid s_i)$ for match states using a Dirichlet mixture regularizer and the weighted counts of the amino acids from the corresponding alignment column.
- Instead of background frequency, or normalizing the relatively few insertion counts, set insertion-state emission probabilities by normalizing the geometric mean of match state frequencies.
- Set transition probabilities based on weighted counts of insertions and deletions in the alignment, plus large pseudocounts based on transitions in many different alignments.

Multi-track HMM*s and secondary structure*

We can also use alignments built using a two-track target HMM:

- Amino-acid track (created with script w0.5 from the SAM-T2K multiple alignment).
- Secondary-structure track (probabilities of {E, H, L} or {E, B, G, H, T, L} from neural net). The correct letters are defined by STRIDE.
- Can align template (AA+2ry) to target model.
- Don't have good way to create template models, nor to align targets to template models.



Secondary structure prediction

For 3-state prediction, neural net is 4-layer:

- 3 hidden layers of 10, 10, 6 units window size 7,11,11
- output layer of 3 units, window size 5.
- trained on SAM-T2K alignments with STRIDE assignment.
- Approximate accuracy $Q_3 = 0.786$, bits/char=0.798 for 3-state prediction. This is among the best predictors in the world (others of comparable quality are PsiPred and J-Pred).

For 6-state prediction, neural net is 4-layer:

- 3 hidden layers of 10, 10, 7 units window size 7,11,11
- output layer of 6 units, window size 7.
- trained on 2752 x-ray structures, SAM-T2K alignments with STRIDE assignment.
- Approximate accuracy $Q_6 = 0.6836$, bits/char=0.968 for 6-state prediction.
- The extra information from 6-state prediction does help (slightly) in fold recognition.

Human input to alignments

- Alignments that scored well examined in 3D, marking aligned residues and identical residues.
- Look for compactness, clustering of identical residues, striping of identical residues across beta sheets, disulphide bridges, ...
- Tweak alignments to improve placement of gaps.

Reversed model for null

- When using the standard null model, certain sequences and HMMs have anomalous behavior. Many of the problems are due to unusual composition—a large number of some usually rare amino acid.
- For example, metallothionein, with 24 cysteines in only 61 total amino acids, scores well on any model with multiple highly conserved cysteines.
- We avoid this (and several other problems) by using a reversed model M^r as the null model.
- The probability of a sequence in M^r is exactly the same as the probability of the reversal of the sequence given M.
- If we assume that M and M^r are equally likely, then

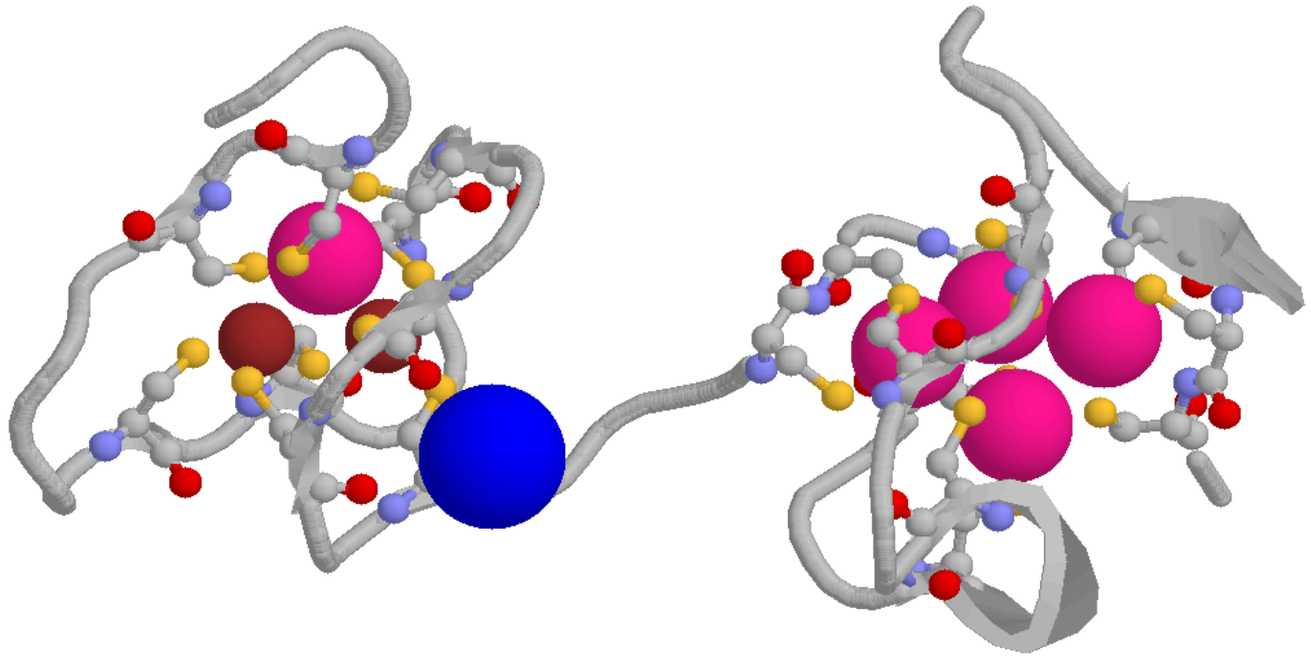
$$\frac{\text{Prob}\,(M \mid S)}{\text{Prob}\,(M^r \mid S)} = \frac{\text{Prob}\,(S \mid M)}{\text{Prob}\,(S \mid M^r)}\ .$$

- This method corrects for composition biases, length biases, and several subtler biases.

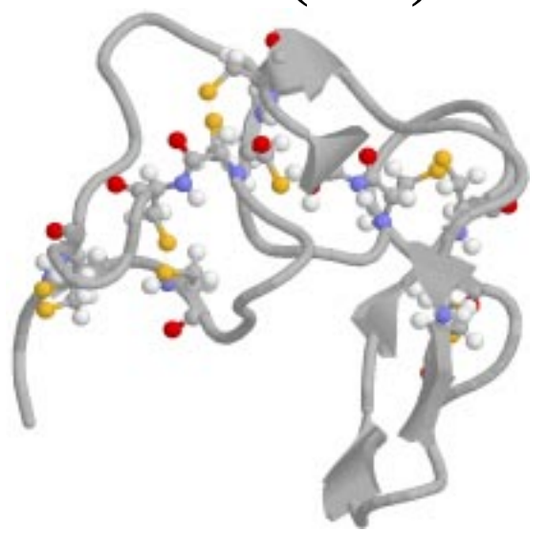
A cysteine-rich protein, such as metallothionein, can match any HMM that has several highly-conserved cysteines, even if they have quite different structures:

		cost in nats	
HMM	sequence	model − standard null	model − reversed-model
1kst	4mt2	-21.15	0.01
1kst	1tabI	-15.04	-0.93
4mt2	1kst	-15.14	-0.10
4mt2	1tabI	-21.44	-1.44
1tabI	1kst	-17.79	-7.72
1tabI	4mt2	-19.63	-1.79

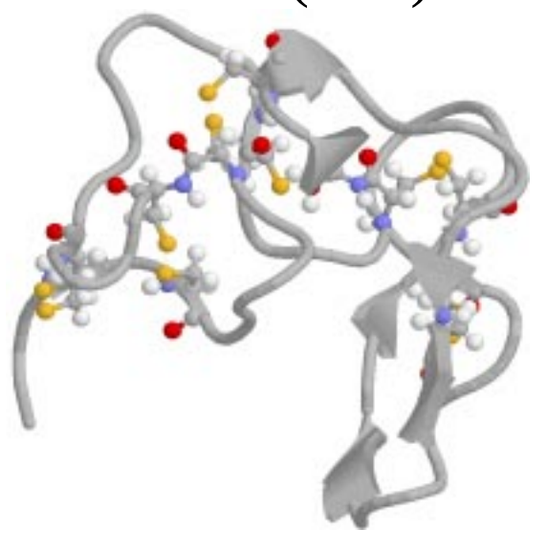
Metallothionein Isoform II (4mt2)



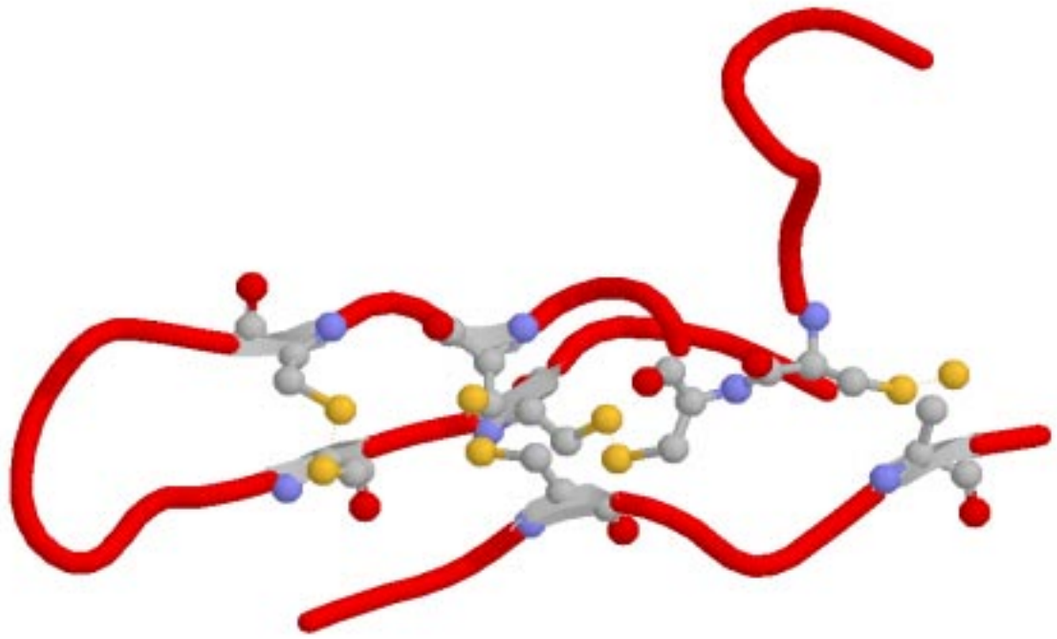
Kistrin (1kst)



Kistrin (1kst)



Trypsin-binding domain of Bowman-Birk Inhibitor (1tabI)

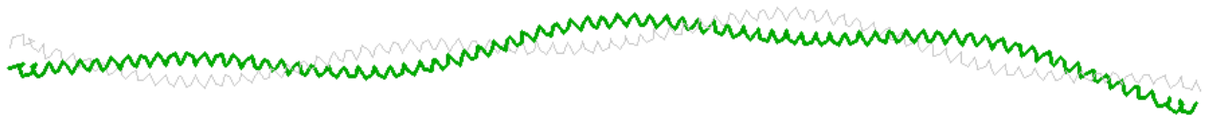


Long helices as source of error

Long helices can provide strong similarity signals from the periodic hydrophobicity, even when the overall folds are quite different:

		cost in nats, normalized using	
HMM	sequence	Null model	reversed-model
1av1A	2tmaA	-22.06	2.13
1av1A	1aep	-21.25	1.03
1av1A	1cii	-13.67	-1.75
1av1A	1vsgA	-7.89	-0.51
2tmaA	1cii	-20.62	0.46
2tmaA	1av1A	-17.96	1.01
2tmaA	1aep	-12.01	0.78
2tmaA	1vsgA	-8.25	0.08
1vsgA	2tmaA	-14.82	-1.20
1vsgA	1av1A	-13.04	-2.68
1vsgA	1aep	-13.02	-3.52
1vsgA	1cii	-11.12	0.28
1aep	1av1A	-11.30	1.79
1aep	2tmaA	-10.73	1.06
1aep	1cii	-8.35	1.38
1aep	1vsgA	-6.87	0.53
1cii	2tmaA	-23.24	-1.48
1cii	1av1A	-19.49	-5.62
1cii	1aep	-12.85	-1.77
1cii	1vsgA	-10.20	-1.57

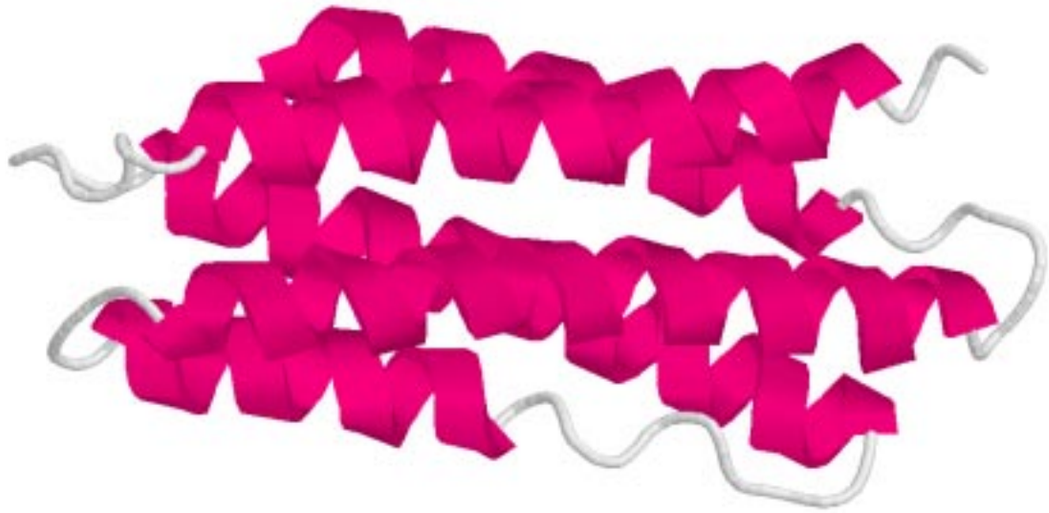
Tropomyosin (2tmaA)



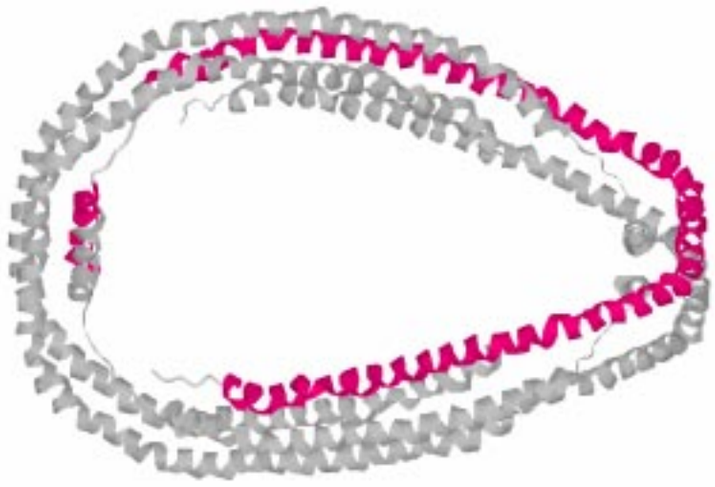
Colicin Ia (1cii)

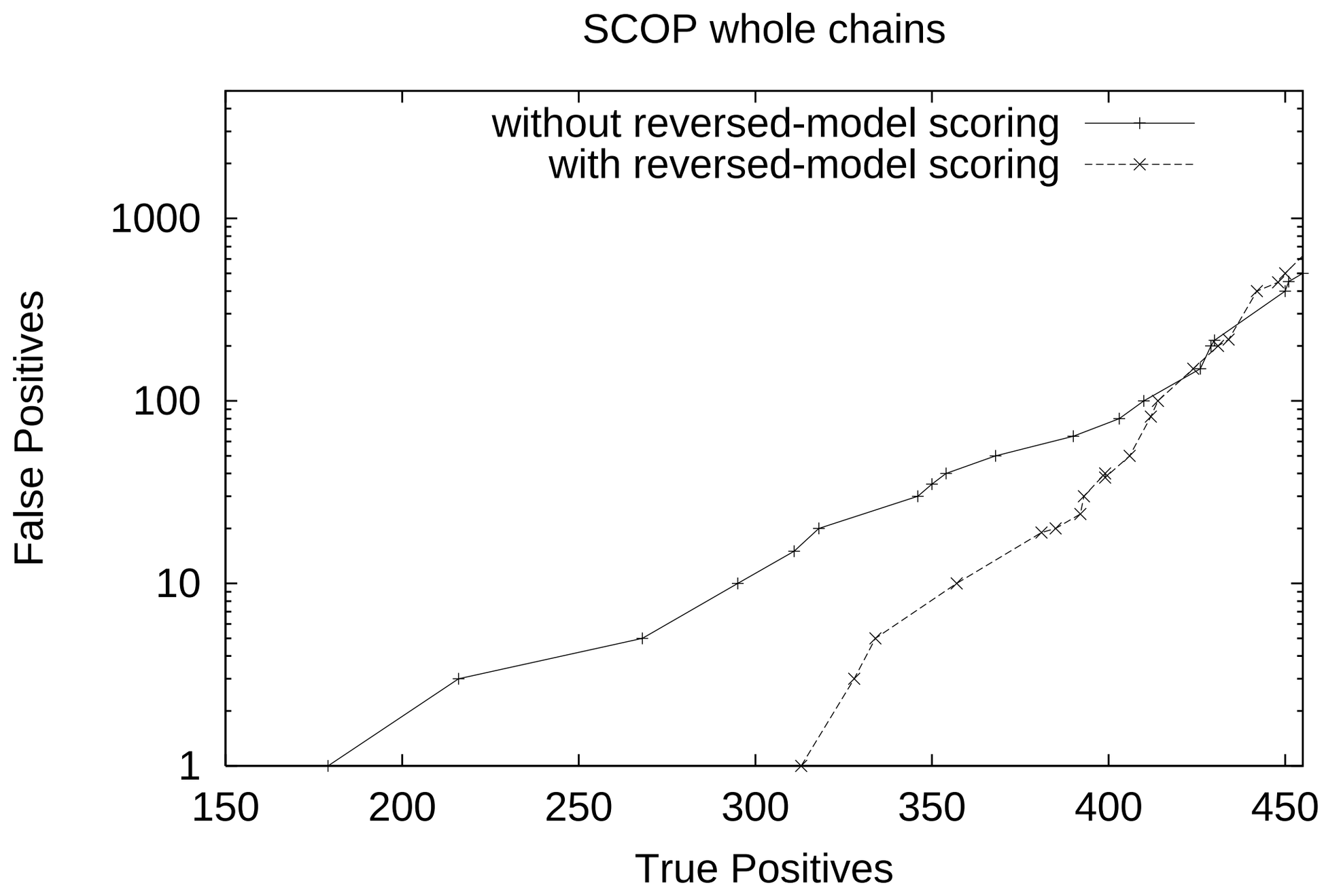
Flavodoxin mutant (1vsgA)

Apolipophorin III (1aep)

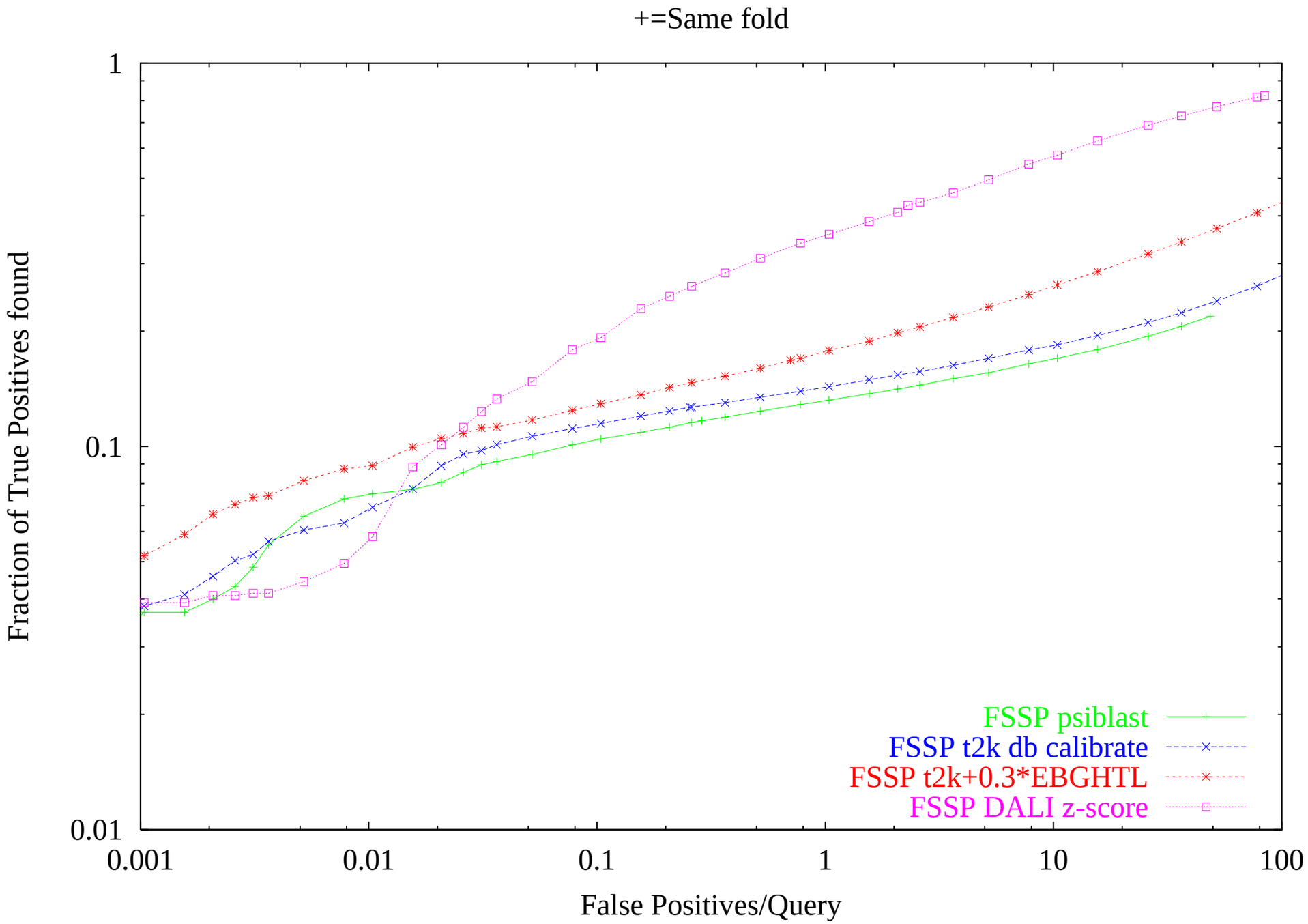


Apolipoprotein A-I (1av1A)




SCOP whole chains
without reversed-model scoring
with reversed-model scoring
False Positives
1000
100
10
1
150
200
250
300
350
400
450
True Positives

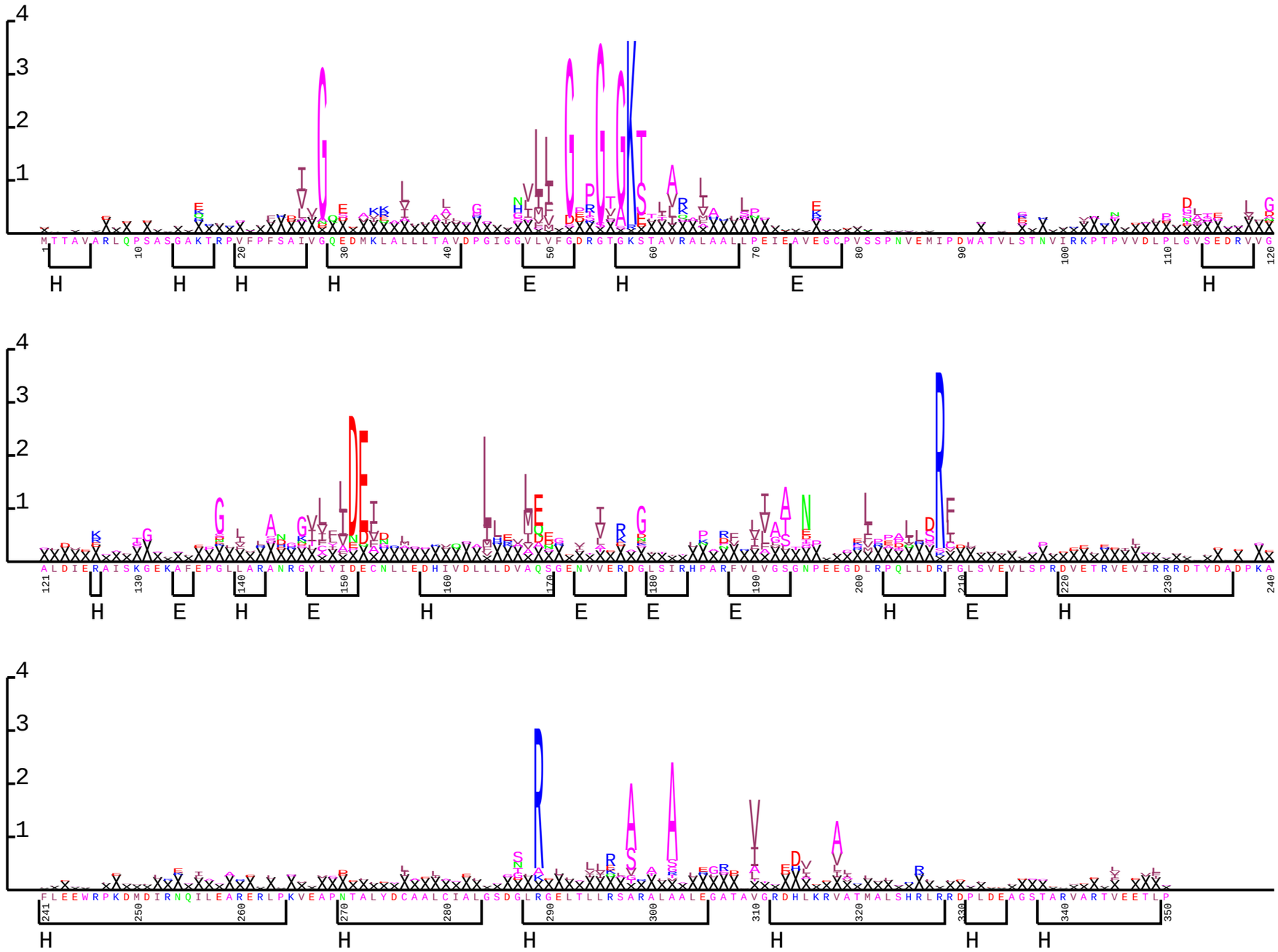
Fold recognition results



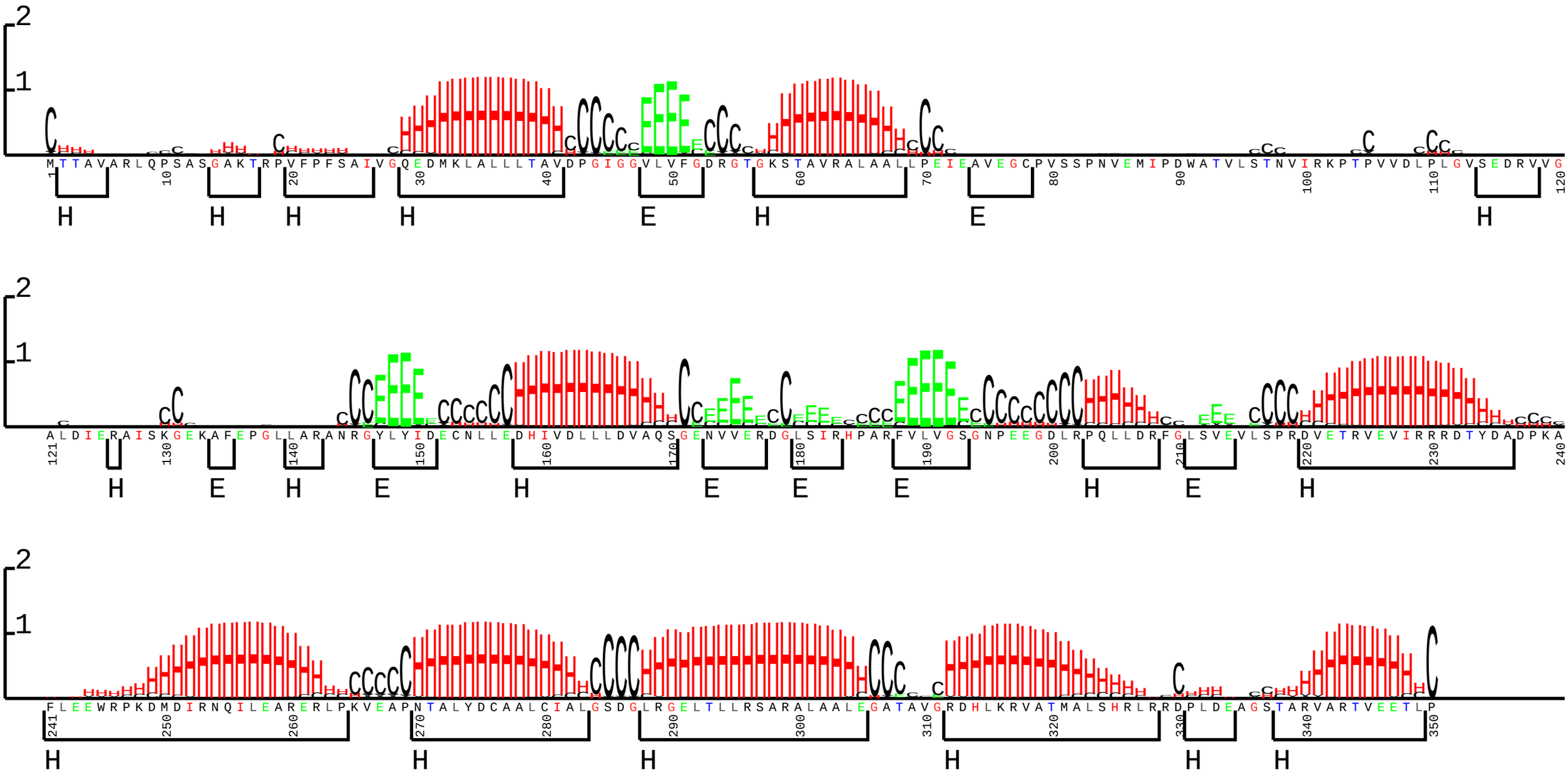
CASP4 example: T0127_2

- SCOP superfamily 3.31.1 with strong scores—like essentially all the automatic servers.
- Best scores with 1do0A (3.31.1.12.5), rather than than 1a5t (3.31.1.12.3), 1d2nA (3.31.1.12.4), 1gky (3.31.1.1.1), or 1nipA (3.31.1.9.3).
- Preference held for both target and template alignments, and both amino-acid only and two-track.
- Second domain, residues 267-350, was better aligned than by other CASP4 predictors.
- Alignment essentially 2-track local alignment. Moved first segment over 1 residue and pulled in final helix from off the end of alignment, based on residue conservation.

CASP4: T0127 sequence logo

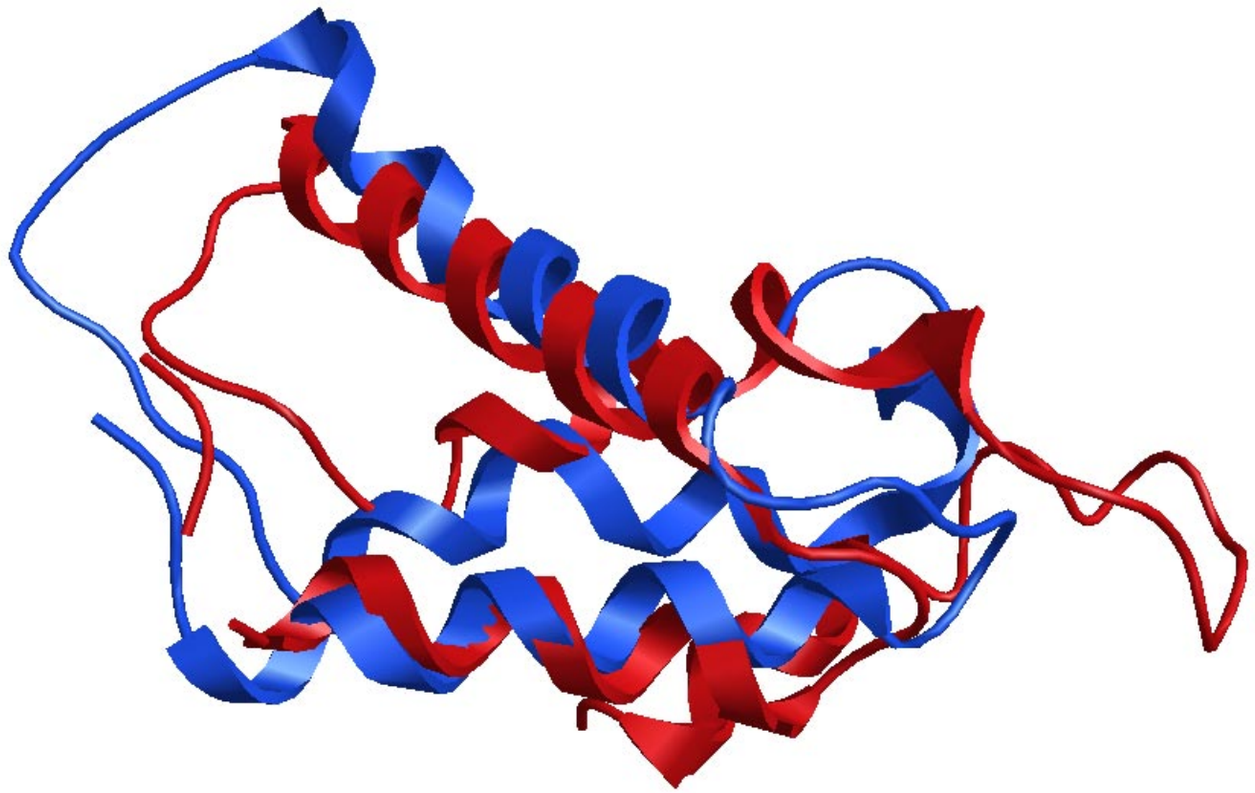


T0127 t2k 2d model



T0127 domain 2 comparison

Predicted=blue solved=red.



UCSC bioinformatics info: `http://www.cse.ucsc.edu/research/compbio/`

SAM tool suite info: `http://www.cse.ucsc.edu/research/compbio/sam.html`

HMM **servers:** `http://www.cse.ucsc.edu/research/compbio/hmm-apps/`

SAM-T99 prediction server: `http://www.cse.ucsc.edu/research/compbio/hmm-apps/T99-query.html`

These slides: `http://www.cse.ucsc.edu/~karplus/papers/combio01.pdf`