A Specific Aims

The goal of this project is to produce the world's best programs for automatic prediction of protein structure from sequence. We will make the tools available to biologists both on the web and as a distributed software package. We hope to increase the number of biologists using our prediction service from the current 700 predictions a week by at least an order of magnitude.

Knowing the structure of proteins is key to figuring out their functions and the mechanisms by which they operate. This information is essential for drug target selection and drug design, as well as for fundamental understanding of both disease processes and the normal operation of cells.

Unfortunately, experimental methods for determining protein structure cannot keep up with the rapid growth in protein sequence data, so computational methods are needed to predict the structure from the sequence data. The computational methods can also aid the experimentalist, as predictions are becoming good enough to use in molecular replacement determination of phases for solving structures by X-ray crystallography, and the ab-initio prediction methods can be combined with limited NMR data to solve structures with much less data than needed with conventional NMR techniques.

This project combines three different, but complementary, approaches to protein-structure prediction: 1D prediction of local structural properties using neural nets, fold-recognition using hidden Markov models, and conformation generation and scoring using fragment packing and an empirical energy function. We are proposing an end-to-end solution that goes from structure to complete 3D models, in a unified framework that will provide state-of-the-art predictions for homology models, for fold-recognition models, and for new-fold models.

An important feature of our new-fold modeling methods is that we do not require the backbone to be contiguous throughout the optimization. For homology modeling, we do not pick a single alignment and freeze the backbone for those residues, but allow many alignments to be sampled and parts of different ones combined. This allows us to represent directly the multiple-segment information that we get from fold-recognition alignments, bringing together the fold-recognition and new-fold techniques.

Because we already represent and manipulate proteins with multiple backbone pieces, handling multimeric protein complexes is a straightforward extension to the existing programs, so we will also apply our method to predicting structure for both homo-multimers and hetero-multimers and not just single proteins.

We will make our programs available free to academics, government research labs, and non-profits, and will provide a free web service for protein structure prediction. This site will return not only 3D structures from a single-sequence input, but also multiple alignments of probable homologs, secondary structure predictions, and sequence logos graphically depicting where the sequence is well-conserved and where secondary structure is confidently predicted. We also plan to write extensive documentation and tutorials to help biologists understand the various outputs we provide and to be aware of the strengths and limitations of the prediction techniques.

B Background and Significance

There are currently three main approaches to predicting protein structure from sequence: 1D predictions, fold-recognition, and new-fold prediction. The literature in the field is voluminous, and so our citations are only representative, not comprehensive.

B.1 1D predictions

These are predictions of some local structural property at each position in a protein chain. *Local protein structure* describes both the environment of an individual residue and its relationship to neighboring residues in three-dimensional space. A *local structure alphabet* is a discrete encoding of one or more properties of local protein structure that clusters residues with similar properties into the same state.

A 1D prediction can be very valuable, both for giving biologists insight into their target protein and for guiding more detailed structure-prediction methods.

The prediction of secondary structure is tested in the world-wide CASP experiments every two years [90, 95, 83] and by an on-going experiment, EVA [29]. Some progress in 3-state secondary-structure prediction has been made in the past several years, but all the top servers (including ours) seem to have similar overall performance.

Since the 1D prediction problem fits easily into standard machine-learning frameworks, many different machine-learning methods have been applied by many different researchers (there are 38 different predictors of secondary structure in the CASP5 experiment [82]). So far, the most successful methods have been ones that use a neural net whose input is derived from a multiple alignment of probable homologs (for example, PhD [105], PsiPred [59], and our SAM-T99 [70]). Many of the methods combine the results from several separately trained predictors, but the improvements resulting from this technique seem to be too small to measure reliably.

Most prediction of local protein structure has centered around a simple three-state classification of *secondary structure* that places a residue in one of three categories: *helix*, *sheet*, or *coil*. This coarse classification provides little information about the coil category that accounts for about 40% of the residues [30]. Although there are many methods for defining fine-grained *alphabets* of local structure [60, 122, 123, 103, 104, 129, 61, 33, 118, 121, 14, 16, 73, 15, 24], there has been very little work exploring whether these alphabets can be used to improve fold recognition or alignments.

It is time to make the problem harder (and more useful) by making more detailed predictions, rather than just 3-state predictions. The main open question is which local properties are most useful to predict.

B.2 Fold recognition

The greatest improvements in protein-structure prediction over the last several years have been in the area of *fold recognition*, which consists of finding a similar protein whose structure has already been determined experimentally (the *template*) and aligning the target sequence to the template.

The improvements have come from several sources: a larger collection of structures in the Protein Data Bank [8], making it more likely that there is a suitable template; a larger collection of sequences in protein sequence data bases, making it easier to find related proteins, thus giving more information about which aspects of the protein sequence are conserved during evolution; better methods for recognizing distant relationships between proteins; and better alignment methods for aligning targets to templates.

Many of the best fold-recognition methods use hidden Markov models (HMMs) as both the search tool and the alignment tool for fold recognition. The HMM techniques were pioneered at University of California, Santa Cruz, and our fold-recognition techniques remain among the world's best [85, 91, 112, 87].

It became clear in 2000, at the CASP4 meeting, that HMMs based only on amino-acid information were no longer competitive with methods that combined amino-acid information with predicted local structure. Indeed, the main value we added to our then-current server (SAM-T99) in our hand-tweaked predictions was the use of experimental two-track HMMs that used predicted secondary structure [70].

Our most recent server (SAM-T02)[1] includes predicted local structure in both the selection of templates and the alignment to templates, and is considerably outperforming the otherwise similar SAM-T99 server in the current LiveBench[2] tests [12, 55].

Open questions include how best to merge local structure information with amino-acid information, what local structure predictions are most useful for choosing templates, and what predictions are most useful for aligning targets to templates.

B.3 New-fold prediction

The most difficult prediction problems are those for which there are no easily-found proteins of known structure to use as templates. These targets can be either novel folds, different from any previously seen, or proteins that have diverged far enough during evolution that even the best fold-recognition methods do not detect a similarity to any known structure.

Closely related to the new-fold problem is *loop modeling*, determining the conformations of regions of a protein that are not present in the templates, or that have very low conservation, and so must be predicted without the benefit of homologous structures.

Both new-fold prediction and loop modeling use similar overall techniques: generation of possible conformations of the protein and scoring of the conformations with a *cost* or *energy* function (generally an empirically-derived one, though some researchers have tried ones based on biophysical principles).

The big problems with new-fold techniques are that the conformational space is huge and that accurate energy functions are very rough, so that almost-right conformations often do not score better than badly wrong conformations.

The main advances in the past four or five years have been in the area of conformation generation. D. Baker and K. T. Simons introduced a new conformation-generation procedure (which they have not given a generic name to but that we refer to as *fragment packing*) that generates locally-reasonable backbone conformations, greatly reducing the size of the search space and reducing the complexity of the energy function [111, 9]. This method showed some promise in 1998 in CASP3 and was the best-performing technique in 2000 in CASP4, both for new-fold models and for loop modeling.

The essence of the fragment-packing method is to copy short stretches (called *fragments*) of backbone conformation from known structures. A small number of possible fragments (say 25) are

[1] http://www.soe.ucsc.edu/research/compbio/HMM-apps/T02-query.html

[2] As of 30 Sept 2002, the SAM-T02 server was doing best of 30 servers on the crucial LGScore1 measure that measures alignment quality (aside from Dali and 3Hit, which are structural aligners given the correct structure, not predictors). It was also doing best on LGScore2, which measures just fold-recognition without considering alignment quality.

selected centered on each residue of the target, and new conformations are generated by randomly selecting a position and a fragment, and replacing the backbone conformation around that position with the conformation from the fragment. This conformational change is very large compared to the sorts of changes used in traditional molecular dynamics and energy minimization programs, but the conformation is locally good, so large portions of conformation search can be sampled quickly.

Other conformation-modification operators can also be defined, such as changing rotamers of the sidechains or making small modifications to phi-psi angles—these sorts of operators are usually applied to make small changes to the conformation to settle into a local minimum of the energy function.

Once conformation-change operators have been defined, the energy function optimization can be done by any of the standard stochastic search algorithms, including simulated annealing and genetic algorithms. One can also do more directed searches that try to improve some component of a score function (such as optimizing a hydrogen bond or removing a clash) by making calculated moves, rather than random moves. When there are many operators, the search procedure generally has to be adaptive, preferentially choosing those operators that are most likely to improve the scoring of the conformations.

Highly detailed energy functions generally provide very rough energy landscapes, with many local minima separated by high barriers. For example, standard Lennard-Jones potentials for Van der Waals contacts give large changes in energy for very small motions of the atoms. These rough landscapes are difficult to search with stochastic search procedures, so most researchers do initial searches with smoother, more approximate energy functions, and do only small conformation changes with more detailed energy functions.

Open problems in this area include choosing good libraries of fragments for a particular target, choosing good conformation-modification operators, designing a good adaptive stochastic-search procedure, and designing good cost or energy functions.

C Preliminary Studies

We have been working in protein-structure prediction since 1996, starting first in fold recognition, then adding secondary structure prediction, and most recently adding new-fold prediction using fragment packing.

Our fold-recognition methods have done well in the biennial Critical Assessment of Structure Prediction experiments since we started (CASP2 [85, 71], CASP3 [91, 67], and CASP4 [112, 70]), and our current server (SAM-T2K) is doing very well in the current round of the ongoing LiveBench experiment [55]. Our secondary structure techniques have done well at CASP3 [95] and CASP4 [83], as well as in the EVA experiment [29]. Our new-fold prediction methods were used seriously for the first time this summer in CASP5, and we are eagerly awaiting the results of that experiment.

The remainder of this section will give descriptions of the techniques we have used—the proposed work relies heavily on extending and improving these methods, so we have dedicated more of the proposal to this section than is common. Much of the work described in this section has not yet been published, though we are submitting papers describing it soon [62, 65].

C.1 Finding and aligning homologs

The heart of both local structure prediction and fold recognition is a multiple alignment of protein sequences similar to the target protein. Most of our effort from 1996 to 2000 went into improving

the techniques for generating this multiple alignment [96, 68, 53, 69, 19].

Multiple protein sequence alignment has been widely used in finding conserved regions in protein families and in predicting protein structures [23, 67, 42, 47, 97, 84].

The quality of the predictions depends critically on the quality of the multiple alignments and the diversity of the sequences aligned, but few of the current multiple aligners are capable of aligning hundreds or thousands of homologous sequences.

Many programs have been developed for multiple protein sequence alignment, and they fall into two classes: progressive and iterative. The classic progressive approach is to build up the alignment gradually by aligning the closest sequences first and then successively adding in more distant ones. Examples include ClustalW [120, 46, 58], PILEUP [40], and PIMA [114]. Another choice is to use an iterative strategy to refine and improve an initial multiple alignment. Programs in the category include PRRP [34], DIALIGN [89], and SAGA [92].

Our method is an iterative hidden Markov model-based search technique, which aligns sequences to a hidden Markov model (HMM) and improves the alignment by retraining the HMM on the sequences. It is close in spirit to the iterative multiple aligners, though rather different in internal implementation.

We use hidden Markov models extensively, and HMMs have been a core component of our research beginning with [44, 76] (see also related work in [5, 117, 26, 27, 4, 11, 41, 25]). HMMs had been used previously in a variety of fields to do discrete time series analysis, most notably in speech recognition [98], but also in DNA analysis [18] and genetic linkage analysis [79]. Following UCSC's seminal work, HMMs are now used extensively in computational biology, primarily to create multiple alignments of protein and other types of sequences, to discover families of related sequences, and to search sequence databases for remote members of these families [96, 68]. They have been used to predict protein structure [71, 67, 70, 31, 51], and to find genes and other functional sites in DNA sequences [77, 78, 100, 13, 56, 127]. The two primary HMM systems for biosequence analysis are the HMMer system, developed by Sean Eddy and others [26], and our SAM system [53].

The structure of the HMMs that are most commonly used in protein modeling is similar to that of a weight matrix or *profile* [38, 36, 10, 45], except that it has specific states and transitions at each position to model insertions and deletions, and the probability parameters for these are different in each position. Such HMMs are called *profile* HMMs. An HMM of this type defines a formal statistical model for sequences in a given protein family, so one can calculate the likelihood of a sequence and find the most probable locations for the insertions and deletions, i.e., the most probable alignment of the sequence to the "consensus model" for the family.

The likelihood is calculated by the *forward* algorithm and the most probable alignment by the *Viterbi algorithm* [124]. Each is a dynamic programming method similar to the Smith-Waterman method used to align two sequences [115]. The forward algorithm can be used to search a database for homologs of the protein family represented by the HMM, and the Viterbi algorithm can be used to create a multiple alignment of all family members. The parameters of the HMM can be estimated from a set of non-aligned family members using an expectation-maximization method known as the *forward-backward* algorithm [98], or by Markov Chain Monte Carlo methods such as Gibbs Sampling, used in [81, 17, 80] (see also [25]), to estimate the parameters of models similar to HMMs. Further details and discussion of the mathematical foundations of HMMs can be found in tutorial articles and text books [98, 25].

Our HMM software had its first general public distribution in 1995 [54, 52]. The Sequence Alignment and Modeling (SAM) software suite has been distributed to about 200 academic sites

and a number of private sites, including 20 fee-based licenses, and has been accessed over 75,000 times, 500–1000 times per week, on our related WWW server.[3]

We pioneered iterative HMM construction from a single sequence and a large non-redundant database. We call these methods the SAM-Txx methods, as they were originally developed as part of our target prediction efforts for the CASP workshops [71, 67, 70]. The SAM 3.2 software release includes SAM-T99, and our web servers also include SAM-T2K (on the T02-query.html page). In our own studies [7, 63, 96, 64] as well as in external studies [88, 35, 87], SAM and the SAM-Txx methods perform consistently better than standard methods and other HMM and profile systems.

SAM-T99 starts with a query sequence (or seed alignment) and searches the non-redundant protein database (NR) using WU-BLASTP [1] to produce two sets of potential homologs: one of very similar sequences ($E < 0.0005$) and one of possibly similar sequences ($E < 300$). The initial WU-BLASTP cull of NR is used for two reasons: we do not expect an HMM built from a single sequence to do any better at finding close homologs than WU-BLASTP, and an HMM database search of all of NR is too slow for building large numbers of alignments. SAM-T2K is similar, but gives a choice of different prefiltering options, and uses a different prefilter threshold for each iteration (by default, it uses prefilter thresholds of 0.01, 1, 10, 400).

Both methods then use four iterations of a selection, training, and alignment procedure. For each iteration they need an initial alignment, a set of sequences to search, a threshold value, and a transition regularizer. From the alignment and regularizer, an HMM is constructed and used to score the set of sequences. All sequences that score better than the threshold value are used to estimate a new HMM. Alignment of the training sequences to that HMM produces the alignment that is the input for the next iteration.

The SAM-Txx methods use sequence weighting for building models from alignments, both internally and when the final alignments are used to create the models for scoring a set of sequences. The relative weights are set with our own weighting scheme which gives more weight to outliers, and the absolute weight is set to get a specific level of entropy averaged over all columns after a Dirichlet mixture regularizer [113] is applied to the weighted counts. The desired entropy is specified as the number of bits saved relative to the entropy of the background distribution. This relative entropy measure has been used previously to characterize substitution matrices [3], and the popular BLOSUM50 and BLOSUM62 matrices corresponds to saving about 0.5 and 0.7 bits per column. The SAM-T99 method uses 0.8 bits per column as the target, but preliminary fold-recognition tests indicated that this may be too high a value, and SAM-T2K uses 0.5 bits per column.

On the first iteration the single sequence passed to the method is used as the initial (trivial) alignment and the close homologs found by WU-BLASTP are used as the search set. The threshold is set strictly (E-value < 0.00001 for SAM-T99 and < 0.0001 for SAM-T2K), so only strong matches to the sequence are considered. Requiring both WU-BLASTP and the initial HMM to score a sequence well ensures that only very similar sequences are included at this stage of the process.

On subsequent iterations the input alignment is the output from the previous iteration and the search set is the larger set of possible homologs found by WU-BLASTP. The thresholds are gradually loosened (E-value < 0.0001, 0.001, 0.01 for SAM-T99, and 0.0002, 0.001, 0.005 for SAM-T2K).

The above selection, training, and alignment procedures consists of several calls to SAM programs. Models are created with SAM's `modelfromalign` program which uses the alignment, sequence weighting, transition regularizer, and Dirichlet mixture to build an HMM. Scoring the

[3] http://www.soe.ucsc.edu/research/compbio/HMM-apps

sequence set with an HMM uses SAM's multiple domain scoring procedure, now part of `hmmscore`, which selects only the portion of a sequence matching the HMM (local scoring [116] as applied to SAM models [119]). From the sequences selected using this procedure, a new model is estimated using SAM's `buildmodel` HMM training program. The alignment of the training sequences back to the resulting HMM is accomplished with SAM's `hmmscore` program. To ensure that the initial sequence to the whole process is not lost, it is added to the training set at this point, and any duplicate sequences in the training set are eliminated.

The SAM-T2K method also differs from the SAM-T99 method in that an extra alignment step is done at the end, aligning the putative homologs with a posterior-decoding alignment rather than a Viterbi alignment. The posterior-decoding alignment is more expensive to run, but alignment tests on difficult examples indicated that it provided slightly better alignments.

We have considered using other multiple alignment algorithms in conjunction with our iterative search, but multiple alignment tests showed that there would be no advantage to using ClustalW [120, 46, 69] and T-coffee [93] was too expensive for the frequently large number of sequences in the multiple alignment.

C.2 Local structure prediction

We have worked on several methods for local-structure prediction, including neural nets, averaged perceptrons, and HMM-based methods. The SAM-T99 web site reports the average of a neural-net method and two quite different HMM-based methods, while the SAM-T02 web site reports neural-net predictions for several different neural nets (and one averaged prediction). All our methods produce a probability vector for each position in the target protein sequence over the alphabet of possible local structure codes.

While other groups have relied on a helix-strand-coil description of secondary structure, we have explored a variety of secondary structure descriptions. We are particularly interested in whether there is a preferred way to encode one-dimensional structure strings with a *local structure alphabet*.

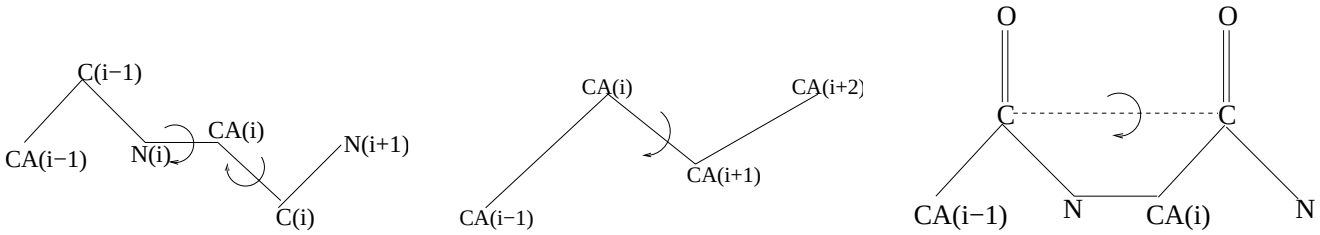
For our purposes, the best local structure alphabets are

- conserved within fold families,
- predictable from amino-acid sequence,
- able to improve template selection, and
- able to improve target-template alignments.

We have developed a protocol for evaluating local protein structure alphabets and have applied the protocol to (so far) nine alphabets of protein backbone geometry. Incorporating predicted local structure information substantially increases the fold-recognition accuracy and alignment quality of our HMM-based methods. Among the local structure alphabets tested, we find that a novel alphabet based on detailed secondary structure states, including classifications of beta strand orientation, is most useful for improving alignments. Fold-recognition improvement is relatively insensitive to choice of a particular local structure alphabet, but we are currently getting the best results with a three-state classification of secondary structure.

C.2.1 Alphabets

For our evaluation experiments, we selected a sample of nine alphabets describing protein backbone geometry. These included the two most widely used secondary structure alphabets (DSSP and STRIDE), three-state reduced versions (STRIDE-EHL and DSSP-EHL), a backbone fragment alphabet



(a) The pair of dihedral backbone angles ϕ and ψ are defined by the atoms C'-N-C_α-C' (ϕ) and N-C_α-C'-N (ψ).

(b) The backbone angle ALPHA is defined for a residue i as the virtual dihedral angle between C_α atoms of residues $i-1$, i, $i+1$ and $i+2$.

(c) TCO is defined for a residue i as the cosine of the dihedral angle around its carbonyl group and that of residue $i-1$.

Figure 1: Three torsion angles used for defining backbone-geometry alphabets.

called PROTEIN BLOCKS (PB) [24], and an alphabet of ϕ-ψ classes (ANG) developed by Bystroff [15] (ϕ and ψ are dihedral angles along the protein backbone [99], shown in Figure 1(a)). We also looked at several novel alphabets. The first is an enhanced version of DSSP, in which we have partitioned the strand class into six classes according to properties of a residue's relationship to its strand partners [62]. Other novel alphabets are based on the backbone angle ALPHA, defined for a residue i as the virtual dihedral angle between C_α atoms of residues $i-1$, i, $i+1$ and $i+2$ (Figure 1(b)) and TCO, defined for a residue i as the cosine of the dihedral angle between its carbonyl group and that of residue $i-1$ (Figure 1(c)).

DSSP We use a seven-letter version of the secondary structure alphabet developed by Kabsch and Sander [60]: E (beta strand), H (alpha helix and pi-helix), T (turn), S (bend), G (3-10 helix), B (short beta bridge) and C (random coil). Assignments are based on patterns of hydrogen bonding.

STRIDE A six-letter (EBGHTC) secondary structure alphabet developed by Frishman and Argos [33]. Assignments are based on ϕ-ψ angles and H-bond energies.

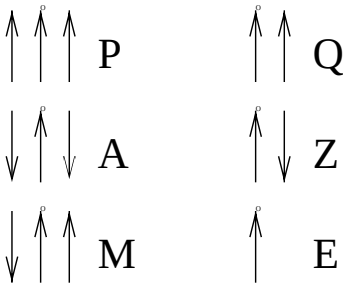
DSSP-EHL, STRIDE-EHL Three-letter (EHL) secondary structure alphabets that reduces DSSP and STRIDE assignments to helix, strand, or coil. In this mapping, G is included in the helix class, B in the strand class, and S and T in the coil or loop class.

PROTEIN BLOCKS An automatically designed alphabet, developed by Alexandre de Brevern and colleagues [24]. This alphabet looks at overlapping residue fragments of length five (chosen empirically), extracted from a non-redundant set of structures, and encodes them as sequence "windows" of ϕ-ψ pairs called *dihedral vectors*. An unsupervised *Kohonen self-organizing map* network [74, 75] was trained on the dihedral vectors with an *RMSDA* (root mean square deviations on angular values) distance metric, to produce a set of 16 clusters, each with a representative Protein Block (PB).

ANG Based on the ϕ-ψ alphabet used in HMMSTR [15]. Bystroff et al. partitioned the ϕ-ψ plane [99] into ten regions, using the *k-means* algorithm [86] on all trans ϕ-ψ pairs in PDB. Cluster

UCSC	Byst	ϕ	ψ
H	H	-61.91	-45.20
G	G	-109.78	20.88
P	B	-70.58	147.22
E	E	-132.89	142.43
D	d	-135.03	77.26
N	b	-85.03	72.26
Y	e	-165.00	175.00
L	L	55.88	38.62
T	l	85.82	-0.03
S	x	80.00	-170.00
not used	c	cis peptide	

(a) Centers of the ten classes in the ANG alphabet with the letters given by us and those originally assigned by Bystroff.



(b) Six letters in the STR alphabet, which expand on the DSSP "E" or strand state. Dots indicate the strand of the residue being assigned. In a beta sheet, this strand is either surrounded by two parallel partners "P", two anti-parallel partners "A" or one anti-parallel and one parallel partner "M". Edge strands (that have only one beta strand partner) have either a parallel partner "Q" or an anti-parallel partner "Z". Finally, we retain the "E" label for strand residues to which DSSP assigns no partners (generally beta bulges).

Figure 2: Definitions for ANG and STR alphabets.

boundaries were calculated with a *Voronoi* method. All cis residues were assigned to an eleventh cluster by Bystroff, but this class was not used in ANG, and the cis residues were distributed among the other 10 classes according to their ϕ-ψ values. Figure 2(a) shows the centers of the 10 classes.

STR An enhanced version of DSSP that subdivides DSSP letter E (beta strand) into six letters (Figure 2(b)), according to properties of a residue's relationship to its strand partners.

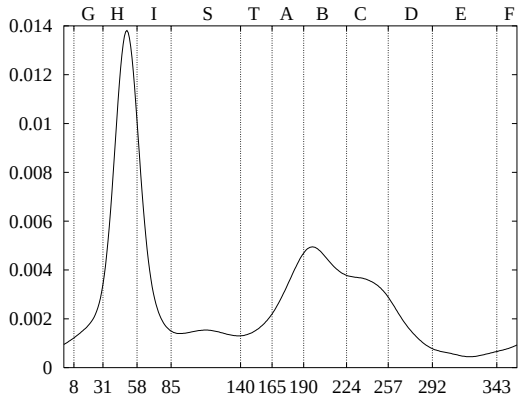
ALPHA We created an eleven-letter alphabet by observing a smoothed histogram of the dihedral angle ALPHA values for all residues in 448 non-redundant high-resolution X-ray structures selected from PDB. We manually assigned breakpoints between ALPHA classes according to location of peaks in this histogram (shown in Figure 3(a)) and separate ALPHA histograms for each of the 20 amino acids.

TCO The TCO alphabet was designed by manually dividing the distribution of TCO cosine values in the `fssp-x` dataset (all x-ray structures that were representatives in FSSP [49]) into four classes (Figure 3(b)). The best centroids for the four classes were selected with the *k-means* algorithm.

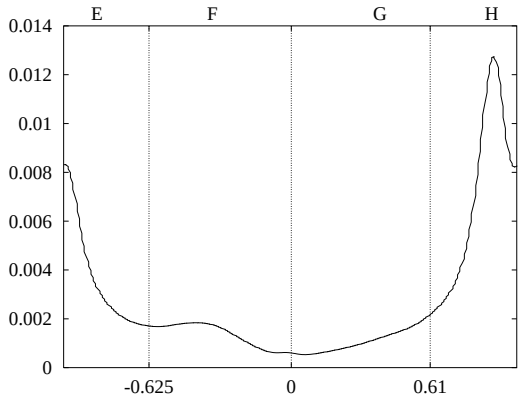
C.2.2 Information content

For each alphabet, we built a structure string, representing each protein chain in our benchmark data set, and added these to our existing library of amino-acid sequences.

We used the library of structure strings to estimate the compositional entropy for each alphabet and the mutual information between all pairs of alphabets (including amino-acid sequence). The letter frequencies in each alphabet are counted and a *confusion matrix* is constructed for each pair



(a) Smoothed histogram of ALPHA angle distribution in monomeric chains from the `dunbrack_cullpdb_pc50_res2.0` data set. The 11-letter ALPHA alphabet is given above the histogram. The peak at letters GHI is for helices, and at ABCD is for beta strands.



(b) Smoothed histogram of the distribution of TCO cosine values. The letter H corresponds to helical conformations, and E to strands.

Figure 3: Alphabet definitions for ALPHA and TCO alphabets.

of alphabets. Each confusion matrix element represents the number of times a letter in alphabet A (l_A) and a letter in alphabet B (l_B) appear in equivalent positions. Alphabet compositional entropy gives an upper limit on our estimate of alphabet conservation, which is based on mutual covariation of letter pairs observed in equivalent positions of a structural alignment. (A simple proof of this bound is given in Durbin et al. [25].) Very low mutual information between an alphabet and the amino acids indicates potential difficulty predicting the alphabet from amino-acid sequences. On the other hand, very high mutual information between two alphabets means that the alphabets carry the same information, so using both would be redundant.

C.2.3 Predictability

In the next step, we ran a test to measure how well structure strings in each alphabet were predicted by feed-forward neural networks. The nets were trained with standard back-propagation and have a four-layer architecture. The same inputs, architecture, and training protocol were used for all neural nets. Only the number of output units in the final layer varied according to the size of the alphabet. We did three-fold cross-validation: the data set was randomly divided into three partitions and, for each alphabet, we trained a net on each two-thirds of the data and tested on the remaining third. The performance of the three nets was averaged and three measures of prediction quality reported: percent of residues correctly predicted to be in one of N states (Q_N), the fractional overlap of secondary structure segments (SOV) in a pairwise alignment of predicted and observed structure strings [106], and the amount of information gained (bits saved) per position in the test set:

$$I = 1/n \sum_{1 \le i \le n} \log_2 \frac{\hat{P}_i(a_i)}{P_\emptyset(a_i)} \ , \tag{1}$$

Name	alphabet size	entropy	MI w/aa	conservation fssp-x mutual info	predictability bits saved per position	QN	SOV
STR	13	2.842	0.103	1.107	1.009	0.561	0.527
PB	16	3.233	0.162	0.980	1.259	0.579	0.542
STRIDE	6	2.182	0.088	0.904	0.863	0.663	0.659
DSSP	7	2.397	0.092	0.893	0.913	0.633	0.610
STRIDE-EHL	3	1.546	0.075	0.861	0.736	0.769	0.733
DSSP-EHL	3	1.545	0.079	0.831	0.717	0.763	0.732
ALPHA	11	2.965	0.087	0.688	0.711	0.469	0.375
ANG	10	2.443	0.228	0.678	0.736	0.588	0.501
TCO	4	1.810	0.095	0.623	0.577	0.649	0.547

Table 1: Summary of the information content, mutual information with amino acid, conservation and predictability of each tested alphabet (see Section C.2.3 for definitions of bits saved, QN and SOV). Conservation (see Section C.2.4) was estimated by calculating the mutual information of letter pairs observed in equivalent positions of FSSP structural alignments. Prediction statistics were computed by three-fold cross-validated testing with four-layer feed-forward neural networks.

where a_i is the correct local structure code for the i^{th} position, $\hat{P}_i(x)$ is the predicted probability of code x in the i^{th} position, and $P_\emptyset(x)$ is the background probability of code x. Note that this measure is independent of the size of the alphabet, unlike more common measures (such as Q_N and SOV) and can be used for comparing results between different alphabets.

C.2.4 Conservation

We evaluate alphabet conservation between fold family members by computing the average *mutual information* of letter pairs observed in equivalent positions of FSSP structural alignments. These alignments each have a "master" belonging to FSSP's "representative set" (none of which share greater than 25% sequence identity) and a collection of "slaves" that are structurally aligned to the master by DALI [48, 49].

For this analysis, we built structure strings for 1609 protein chains taken from the FSSP representative set and also for their slaves that aligned to the master with DALI Z-scores ≥ 7.0. Because many sets of slaves are large and redundant, and because we are interested in conservation of properties between sequences with very low amino-acid similarity, we thinned the alignment so that no two sequences had identity $> 20\%$.

Finally, we extracted the ranges of all structurally conserved regions in the FSSP alignments and constructed tables of aligned residue positions. For all possible pairs of letters (X, Y) in a local structure alphabet, we counted the number of times X in the master structure was paired with Y in a slave structure in an aligned position, and computed their mutual information:

$$M(X;Y) = \sum_{X,Y} P(X,Y) \log_2 \frac{P(X,Y)}{P(X)P(Y)} \tag{2}$$

where probabilities were estimated by normalized counts of letters in the dataset. Since estimates of mutual information based on small samples have been shown to be artificially inflated [126, 125],

we perform a *small sample correction* on $M(X;Y)$. The small sample effect is evidenced by the fact that when aligned letter pairs are scrambled, a non-zero mutual information is still measured between randomized pairs. By repeatedly scrambling the aligned letter pairs, we can compute a distribution of "random mutual information" and correct for the over-estimate by subtracting the mean of this distribution from $M(X;Y)$ [21].

Table 1 presents our analysis of the compositional entropy, mutual information with amino acid, conservation, and predictability for the alphabets studied. The alphabets are ranked according to their conservation in FSSP fold families. According to this data, the STR alphabet best encodes conserved properties of local backbone structure, followed by PB, STRIDE, and DSSP. Because STR is an expansion of the DSSP alphabet, its being more conserved shows us that the properties of strand pairing described in STR have been preserved in remote homologs.

The bits saved (information gain) measure of predictability does not depend on alphabet size and is strongly correlated with alphabet conservation (r=0.84). We have found this to be the most useful measure of predictability when comparing different alphabets.

C.3 Fold recognition

Structural information about available templates has been shown to improve performance of both profile and threading fold-recognition methods [107, 32, 101, 22, 72]. In this section, we evaluate the results of enriching HMMs, built with the SAM software [52, 53], with frequency profiles derived from predicted one-dimensional structure strings. The models are designed to search a template database for structurally similar remote homologs and to align target-template pairs.

There have been previous attempts to improve fold recognition and target-template alignments by incorporating predicted secondary structure information into profile HMMs [32, 43], but our approach differs in several important ways. Other groups score the sequence and/or predicted secondary structure of a target protein against a template library of HMMs. Their HMMs were trained on selected sequences, sharing a common SCOP family (clear evolutionary relationship) or fold (major structural similarity) [50], and the known secondary structure of these sequences was incorporated into model training. One group used only secondary structure sequences to build their library [32], while the other trained on both amino-acid and secondary structure sequences [43].

We have recently extended our SAM software to handle *two-track* profile HMMs, in which each match node contains emission probabilities of predicted local structure information, in addition to amino-acid emission probabilities [70, 66]. As shown in Figure 4, the primary *track* of amino-acid emission tables is built with the SAM-T2K iterated seed alignment algorithm, and the secondary track of local structure emission tables is estimated by a neural network.

Figure 5 shows a graphical depiction of a small two-track HMM.

For our fold-recognition experiments, we used the SCOP database as a standard for identifying related proteins. We created a benchmark of non-homologous, protein whole chains. Our study focuses on proteins that are difficult to detect by sequence similarity, so we excluded proteins sharing greater than 20% sequence identity from the benchmark set. We define correct hits as proteins which share a common SCOP fold (major structural similarity).

We wanted to see if including local structure information in our HMMs would improve fold recognition, and which of the local structure alphabets gave the best results. We built one PSIBLAST profile (using four iterations with threshold set at 0.0005) [2, 109], one amino-acid-only HMM, and eight two-track HMMs, for each chain in the benchmark set, trying each candidate local structure

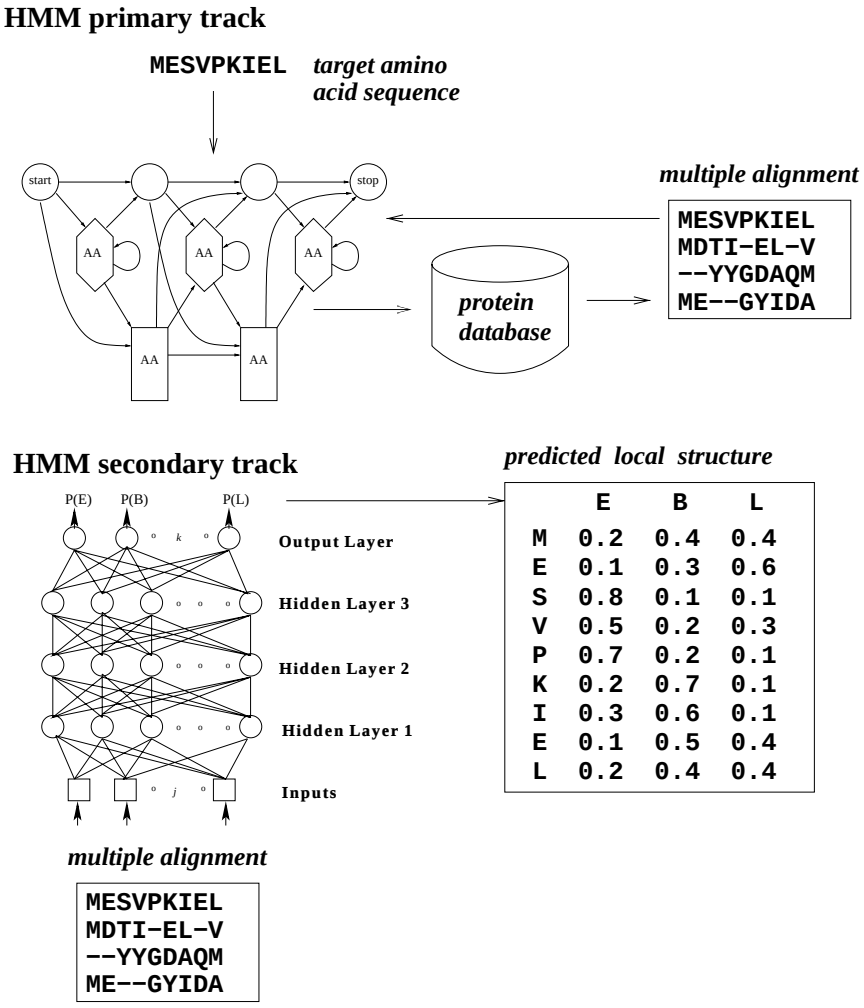


Figure 4: The 2-track HMM has a primary track of amino-acid emissions and a secondary track of local structure alphabet emissions. The primary track is constructed with the SAM-T2K iterated seed alignment algorithm. The secondary track is modeled with predicted local structure probabilities that are estimated by a neural network. Input to the neural net is the final multiple alignment generated by SAM-T2K.

alphabet as a secondary track. Each benchmark profile and HMM was used to rank all chains in the set according to E-values of the PSIBLAST or HMM scores [65]. On the basis of a small number of experiments, we chose weights of 1.0 for amino-acid emissions and 0.3 for local structure emissions, when computing the two-track HMM scores. Results for each alphabet could probably be improved by optimizing these parameters.

In this setting, we define results of a *query* as the list of E-values received by all chains in the benchmark set, with respect to a single profile or HMM. Performance of a candidate alphabet was evaluated with respect to all benchmark HMMs built with the alphabet as a secondary track (1298 queries). We report fold-recognition performance of each method both in terms of percent of true positives recognized vs. false positives per query, and in terms of ROC curves and ROC numbers.

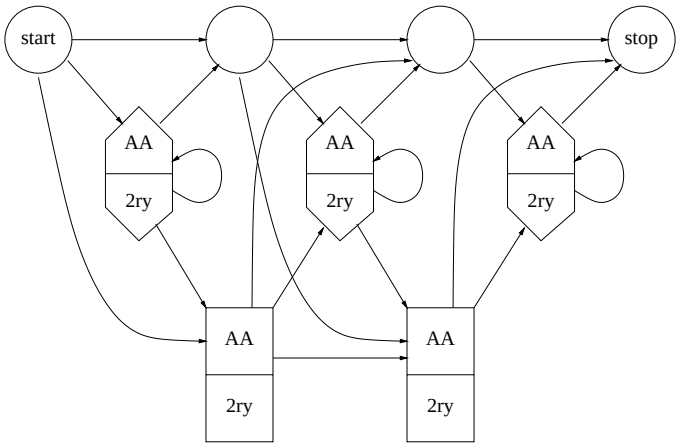


Figure 5: This picture shows how the two-track SAM-T2K target HMM is organized. The "AA" and "2ry" labels in the boxes refer to emission-probability tables for amino acids and local structure labels, respectively [70]. The only change from the profile HMMs used previously with SAM is the addition of predicted emission probabilities for a local structure alphabet to the match states—the insertion states get background probabilities for local structure and the transition probabilities are identical to the amino-acid-only HMMs. A real profile HMM has as many match states as alignment columns in the multiple alignment.

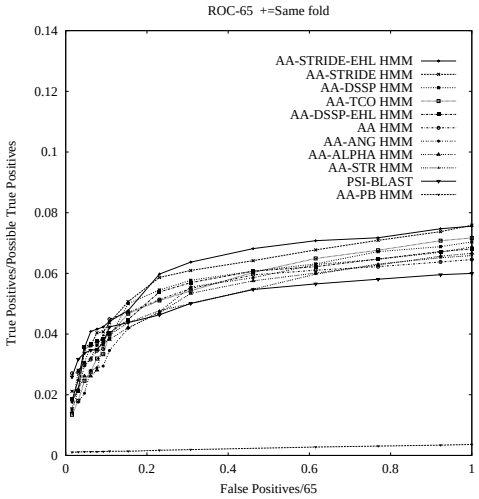
C.3.1 ROC numbers

The ROC curve is a plot of true positive fraction vs. true negative fraction using a sliding threshold and provides an accurate, quantitative measure of both the sensitivity and specificity of a database search.

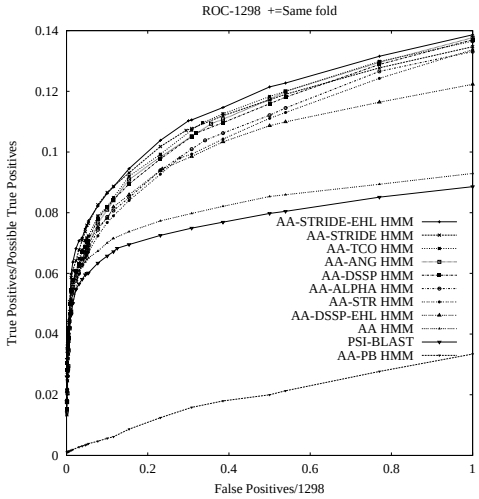
The total area under the ROC curve gives the probability of a correct classification [37]. Because of the very large number of true negatives in a typical database search, the area is usually calculated under a truncated ROC curve, with a fixed *ROC number* (ROC_N), where N is the number of true negatives used in the calculation.

Our fold-recognition results are shown in Figure 6. There is no clear separation between multi-track and single-track HMMs in the very low false positive range (0.05 false positives per query). However, if we are willing to accept between 0.1 to 1.0 false positive per query (a reasonable threshold, given the low homology of the proteins in the test set), the two-track models increasingly recognize more correct folds with fewer false positives than either PSIBLAST or SAM-T2K amino-acid HMMs. The exception is PB, which in spite of high compositional entropy and predictability, did very poorly at fold recognition. This anomaly is an artifact of the reverse-sequence null model SAM uses to compute HMM scores [65].

Table 2 shows the fold-recognition ROC_{65}, ROC_{130}, ROC_{649}, and ROC_{1298} numbers (see Section C.3.1) for two-track HMMs, single-track HMMs, and PSIBLAST. These were computed by estimating the area under the corresponding ROC curves with a trapezoidal method. The thresholds correspond to 0.05 FP/Q (false positives per query) at ROC_{65}, 0.1 FP/Q at ROC_{130}, 0.5 FP/Q at ROC_{649}, and 1.0 FP/Q at ROC_{1298}. The addition of backbone geometry information in the SAM-T2K HMMs clearly improves fold-recognition performance, but choice of alphabet for the secondary track does not make much of a difference. Although the two-track STRIDE-EHL HMM has the best ROC numbers, the advantage is very small over other backbone-geometry alphabets.



(a) ROC_{65}. Number of negatives recognized fixed at 65 (0.05 false positives per query).



(b) ROC_{1298}. Number of negatives recognized fixed at 1298 (1.0 false positives per query).

Figure 6: Results of three-fold cross-validated fold-recognition tests on our benchmark dataset shown for two-track HMMs with an amino-acid primary track and STRIDE-EHL, STRIDE, DSSP, DSSP-EHL, TCO, ANG, ALPHA, STR, and PB (Protein Blocks) on the secondary track, amino-acid only HMMs, and PSIBLAST run with four iterations. The methods are ranked in the legends according to ROC_N score. As shown in Figure 6(a), AA-STRIDE-EHL HMMs have the best accuracy by a small margin. With looser thresholds, in Figure 6(b), the accuracy of all the local structure HMMs is significantly better than that of the AA-only methods. The poor performance of AA-PB HMMs is an artifact of the reverse-sequence null model SAM uses to compute HMM scores [65].

Method	ROC_{65}	ROC_{130}	ROC_{649}	ROC_{1298}
AA-STRIDE-EHL HMM	**0.0632**	**0.0724**	**0.1010**	**0.1155**
AA-STRIDE HMM	0.0615	0.0716	0.0985	0.1125
AA-DSSP HMM	0.0579	0.0673	0.0952	0.1113
AA-TCO HMM	0.0571	0.0673	0.0970	0.1126
AA-DSSP-EHL HMM	0.0567	0.0654	0.0906	0.1032
AA-ANG HMM	0.0547	0.0650	0.0955	0.1119
AA-ALPHA HMM	0.0537	0.0635	0.0914	0.1078
AA-STR HMM	0.0533	0.0625	0.0901	0.1065
AA HMM	0.0552	0.0612	0.0755	0.0823
PSIBLAST	0.0514	0.0572	0.0708	0.0776
AA-PB HMM	0.0024	0.0035	0.0122	0.0196

Table 2: ROC numbers (see Section C.3.1) from three-fold cross-validated fold-recognition tests on a difficult set of 1298 whole chains. The numbers are an approximation to area under the ROC curve for thresholds of 65, 130, 649 and 1298 negatives recognized. The thresholds correspond to 0.05 FP/Q (false positives per query) at ROC_{65}, 0.1 FP/Q at ROC_{130}, 0.5 FP/Q at ROC_{649}, 1.0 FP/Q at ROC_{1298}.

Reference alignment	Difficult set mean shift score DALI	CE	Moderate set mean shift score DALI	CE
two-track t2k STR	0.320	**0.307**	**0.466**	**0.418**
two-track t2k PB	0.309	0.303	0.435	0.395
two-track t2k DSSP	0.306	0.295	0.454	0.402
two-track t2k STRIDE	**0.357**	0.292	0.452	0.400
two-track t2k STRIDE-EHL	0.298	0.290	0.438	0.396
two-track t2k DSSP-EHL	0.297	0.287	0.435	0.391
two-track ALPHA	0.288	0.279	0.429	0.387
two-track ANG	0.286	0.276	0.422	0.407
two-track TCO	0.284	0.276	0.421	0.374
one-track AA	0.220	0.219	0.365	0.325
one-track AA FSSP seed	0.219	0.192	0.415	0.330

Table 3: Evaluation of alignment quality for a difficult set of 200 protein pairs with high structural similarity but low sequence identity (3-24%) and a moderately difficult set of 340 protein pairs. Mean shift scores [19, 20] are shown for alignments done with single-track SAM-T2K amino-acid HMMs, single-track amino-acid HMMs trained on FSSP structural alignments, and two-track SAM-T02 HMMs with several different local structure alphabets.

C.3.2 Alignment

We used two sets of protein pairs to evaluate alignment quality: a difficult set of 200 pairs, with high structural similarity but low sequence identity (3-24%) and a moderately difficult set of 340 pairs (homology detectable by SAM-T2K HMM or PSIBLAST but not by BLAST).

In both test sets, two local alignments were produced for each pair, by building a SAM-T2K HMM for one pair member and aligning the other to the model using the Viterbi algorithm [124]. We tested SAM-T2K amino-acid-only HMMs, HMMs built with an FSSP structural alignment as the seed, and eight types of SAM-T2K two-track HMMs, in which one of our eight candidate local structure alphabets was used for the secondary track. Track weights were set at 1.0 for the amino-acid and 0.3 for the secondary track, as in the fold-recognition tests.

The resulting alignments were compared to structural alignments of the same pairs. To avoid possible bias, two structural alignment methods were used in the analysis: DALI [49] and CE [110]. The mean *shift score* [19, 20] between alignments produced by the HMMs and by the structural aligners was computed. Shift score measures the disagreement of two alignments, typically a predicted *candidate* alignment and a *reference* structural alignment of the same sequence pair. It quantifies several kinds of alignment error in a single number: misalignment, aligning too much, and aligning too little, and it compares well to accepted measures, such as *alignment specificity* and *alignment sensitivity*. It is 97% correlated with *percentage of residues aligned correctly*, but also incorporates information about alignment length, shift error, and coverage [19, 20]. The range of shift score is from -0.2 (worst) to 1.0 (best), achieved only when two alignments are identical.

Table 3 shows results of our alignment quality tests on both test sets.

Previous to this work, our best quality alignments were produced by amino-acid HMMs trained

on an initial FSSP structural alignment.[4] As shown by the reported mean shift scores, the two-track HMMs produce better quality alignments than SAM-T2K amino-acid-only HMMs and the FSSP-seeded amino-acid-only HMMs. Results are reasonably consistent, regardless of which structural aligner is used as reference.

On the moderately difficult set, the two-track STR-HMMs produce the best quality alignments, as measured by mean shift score, with respect to both DALI and CE structural alignments of the same pairs. On the difficult set, the STR HMMs produce the best alignments when compared to CE, but STRIDE-HMMs do better when compared to DALI. Overall, the highly informative, 13-letter STR alphabet, which encodes detailed predictions about beta-sheet structure, is the best choice for HMM alignment. However, when there is very low sequence identity between a pair of proteins, the 6-letter STRIDE alphabet may be a better choice.

We have shown that adding predicted local structure information to profile HMMs can improve detection and alignment of structurally similar proteins, even when there is very little sequence relationship. A simplified helix-strand-coil representation of secondary structure works well for fold recognition, but not so well for alignments, where greater benefit is found by using a more detailed alphabet of local protein structure. The high degree of conservation and predictability in the STR alphabet suggests that there is useful information in the patterns of strand orientation found in beta sheets. The success of this alphabet at aligning structurally similar proteins with low sequence identity shows that these patterns are both predictable from amino-acid sequence and conserved in remote homologs.

We believe that highly informative, detailed alphabets like STR and PB have the potential to do better fold recognition than simple alphabets like STRIDE-EHL and TCO, and that best results will be achieved by combining several alphabets that encode different kinds of information about local protein structure. We are currently investigating ways to create superior null models for use with many-track HMMs and HMMs with non-reversible alphabets such as PB (see Section D.1.4).

C.4 New-fold prediction

Since CASP3 in 1998, when David Baker's program Rosetta showed some promise for ab-initio prediction [111], we have been working on a fragment-packing program named *undertaker* (so named because it optimizes residue burial). We tried a version of the program on a couple of targets in CASP4, when the program was far from complete, but, as expected, the results were poor.

A major effort was made in spring 2002 to speed up undertaker, fix major bugs, add new cost functions, and add new conformation-change operators. It was barely finished in time to use it in CASP5, and still requiers extensive testing and tuning. The structures it generated for CASP5 targets contained common super-secondary structure motifs and had a packing density not much lower than that of experimental structures, so we have made some progress in our conformation generation and scoring, even if we cannot yet quantify the improvement.

One major addition to the SAM suite of HMM tools was a fragment-finder program, `fragfinder`, that looks for short matches between an HMM and template sequences, to provide a library of medium-length fragments for the conformation generation. In preliminary tests, we found that our

[4] We have never gotten good fold-recognition sensitivity with FSSP-seeded HMMs, a result also reported in a recent study [39], which attributed this to the relatively small size of structural alignments. We believe that the high diversity of the sequences in a structural alignment also contributes to loss of signal in the HMM.

2-track HMM using the STR alphabet produced better fragments (a higher percentage of low RMSD fragments) than an amino-acid-only HMM, but have not done sufficient testing to claim that this is generally true.

This section will sketch out a few of the main ideas of undertaker, which can be viewed as consisting of four main parts: the conformation representation, the conformation-changing operators, the stochastic-search method, and the cost function.

C.4.1 Conformation representation

Selection of a conformation representation and data structure is critical to efective fragment packing, as it affects the computation time, the possible conformation-change operators, and the possible cost functions. We represent protein conformations as the 3-d coordinates for all heavy atoms (not hydrogens). Using a full 3-d representation for all heavy atoms, rather than a more compact one such as ϕ-ϕ angles or sidechain centroids, slows down conformation generation slightly, but allows much more flexibility in defining cost functions. One decision we plan to revisit is whether to include explicit hydrogens—having explicit hydrogens would make hydrogen-bond scoring simpler, but increases the size of the conformational space, since torsion angles for the NH_3 and OH groups would then need to be set.

We do not require the backbone to be contiguous, but allow breaks between residues. This allows us to represent directly the multiple-segment information we get from fold-recognition alignments, bringing fold-recognition and new-fold techniques into a unified framework. For homology modeling, unlike Rosetta [9], we do not pick a single alignment and freeze the backbone for those residues, but allow many alignments to be sampled and parts of different ones combined.

Allowing broken backbones introduces a problem that is not present in programs, like Rosetta, that use a contiguous backbone: what is the relationship between unconnected parts of the backbone? what moves when a fragment is inserted?

To solve this problem, we represent the protein as a tree of *segments*, where each segment is a contiguous piece of the protein with properly formed peptide bonds. The edges between segments, called *tertiary edges*, indicate which pairs of atoms are thought of as holding the segments together. When a conformation is changed, say by fragment insertion, the tree is broken into subtrees and the subtrees rigidly transformed. Note that a subtree may consist of widely separated parts of the protein chain, as would be necessary for holding together a domain while an inserted other domain changes shape. The tree structure is generated and maintained automatically as the conformation is read in and manipulated.

C.4.2 Conformation-changing operators

We have implemented several conformation-changing operations, beginning with the fragment insertion operation introduced by Simons and Baker [111]. We have three sources for fragments to insert: very short ones (1-4 residues) from a generic fragment library, which must match exactly on all residues, medium-length ones (9-12 residues) found by `fragfinder`, and variable-length ones that come from fold-recognition alignments. We also have an operator for inserting two fragments simultaneously, to allow for hinge-like motions of part of the conformation, though there is currently no constraint that the conformation change actually be hinge-like.

In addition to fragment insertion, we have alignment insertion, which inserts several segments at once, keeping them in the same frame of reference. This operator allows us to import complete fold-recognition results into our fragment-packing optimization.

We have a number of operators that attempt to improve some part of our cost function—reducing breaks, forming or improving disulfide bonds, reducing clashes, reducing the cost of user-specified constraints, and so forth. Many of these operators work by trying a small number of potential fragment insertions and computing the effect the fragment insertion would have on just part of the cost function, selecting the fragment insertion that appears to make the most improvement.

We also have operators for repositioning subtrees, either jiggling them a small amount or trying to find the optimal placement for them, with various ways of splitting the tree into subtrees. On a smaller scale, we also have operators for changing the rotamers of the residues without changing the backbone, to improve packing or reduce clashes.

Since we are using a genetic algorithm for our stochastic search, we also have crossover operators that combine parts of two conformations to get a new one.

C.4.3 Stochastic search

As mentioned above, we are using a genetic algorithm to search conformational space. We start from a set of conformations (randomly, based on fold-recognition alignments, or from previous runs of undertaker), and randomly apply operators to generate new conformations. New conformations that score well are added to the pool for the next generation, and poorly scoring older conformations are eliminated. To make sure that the pool mixes rapidly, we keep no more than 40% of the conformations from the previous generation.

We keep track of the success rate for each operator (how often it results in a conformation being kept in the pool) and adjust the probability of applying the operators based on their success. The adaptation scheme we are currently using is rather crude and sometimes gets stuck applying only one or two of the operators, if it has initial success with them.

We use the results of several runs of the genetic algorithm to seed the pool for another run, often getting noticeable reduction in cost from crossover between the local minima.

C.4.4 Cost function

Substantial effort was put into making the cost function easily modified and extended, as it was quite clear that a lot of future work would be on different scoring functions.

The cost function can be defined at run time as a linear combination of any subset of a large number of different basic cost functions, and the basic cost functions themselves can be parameterized at run time. We currently have over two dozen basic cost functions and still have several more that we believe we should implement and test. New basic cost functions are very easily added to the code, and they add no computational cost unless they are specifically requested in the linear-combination specified at runtime.

One of our most important cost functions, indeed the one that gives undertaker its name, is the *burial* function. This is a parameterized function that counts for each residue the number of atoms of the conformation within a sphere near the residue and scores the sphere based on the probability of seeing that number of atoms. The sphere is referred to as a *spot*, and the number of atoms whose centers are within the sphere as the *burial* of the spot. The parameter files for a burial function include a specification of where the center of the sphere is relative to the residue, the size of the sphere, and the smoothed probability distribution of burial for each residue type. The undertaker program includes functionality for optimizing the spot locations (to maximize burial for *dry spots*, to minimize burial with a distance constraint for *wet spots*, or to maximize mutual information with

the residue type for *generic spots* whose location does not depend on the type of residue) and for collecting and smoothing histograms for defining the probability distribution.

We also have basic cost functions that can accept the predicted probabilities over a local structure alphabet for a target and score the conformation using them (currently working only for the ALPHA torsion-angle alphabet).

One important basic score function accepts user-specified distance constraints on pairs of atoms, and tries to satisfy these constraints while generating conformations. These constraints can come either from educated guesses by the user of the program or from experimental data (such as NMR experiments or cross-linking experiments).

D Research Design and Methods

We propose two major research directions: improving our fold-recognition and alignment techniques and improving our new-fold techniques. We also propose a documentation and outreach effort, to increase the use of our tools by bench biologists.

D.1 Fold-recognition and alignment

Our HMM-based fold-recognitions and alignment methods are fairly mature, but we have identified several areas in which they can be improved:

D.1.1 Local structure prediction

We do not anticipate major improvements in our *methods* for predicting local structure, but will clean up the software to make it distributable to other researchers. The main thrust of the research here is in choosing which properties to predict.

We have just begun our exploration of which local structure alphabets are worthwhile to predict. So far we have looked at a representative sample of backbone-geometry alphabets—we plan to examine other backbone-geometry alphabets and other structural properties, such as various measures of solvent accessibility or burial. We will evaluate alphabets from the literature as well as new ones based on our own ideas.

D.1.2 Multi-track HMMs parameters

Our initial tests rather arbitrarily set the weight for the amino-acid track at 1.0 and for the local-structure track at 0.3—we need to do systematic explorations of the effects of varying these weights, to find a heuristic for choosing the weights for new alphabets. We will also investigate using more than two tracks on our HMMs, particularly once we have significantly different local structure alphabets, such as secondary structure and burial. Balancing the weights for the different alphabets will require considerable experimentation, with both alignment tests and fold-recognition tests.

D.1.3 HMM-HMM alignment

Currently, our methods rely on aligning sequences (or pairs of amino-acid and secondary structure sequences) to HMMs, but we generate HMMs for both the target and the template proteins, and an HMM-HMM alignment technique might help us improve our fold-recognition and alignments.

Some fold-recognition techniques (such as FFAS [57]) have been using profile-profile alignment with good results, and recent research has pointed out potentially better profile-profile alignment

methods [128, 28]. We plan to extend these profile-profile techniques to HMM-HMM alignment techniques and to extend them to multi-track HMMs.

D.1.4 Better null models for multi-track HMMs

We have been using reversed-sequence null models for the past several years for our HMMs, and they worked very well for amino-acid-only HMMs but have occasionally failed badly for multi-track HMMs. For example, we traced the poor performance of the protein-blocks alphabet in our fold-recognition tests to the fact that real sequences over the protein blocks alphabet are drawn from a distinctly different distribution than reversed sequences. This behavior violates an important assumption of the reversed-sequence null model, the *reversibility property*: that scores for randomly selected sequences and reversed sequences come from the same distribution.

Even when alphabets individually satisfy the reversibility property, a combination of them may not. For example, though both the STRIDE and ALPHA alphabets satisfy (approximately) the reversibility assumption, the cross-product of them does not.

One approach we will examine is using first- or second-order Markov chains to model sequences in the non-reversible alphabets, and use these Markov chains as null models for the HMMs.

Our current calibration of the statistical significance of our HMMs uses an ad hoc family of distributions to fit the fat tails observed in the scoring. The calibration method relies on the symmetry of the distribution produced by the reverse-sequence null models, which in turn relies on the reversibility property. When we develop new null models for non-reversible alphabets, we will also have to develop new calibration techniques for determining the E-values.

D.1.5 Trimming alignments

We have studied improving HMM-generated alignments by trimming away low-reliability regions [19], but have not yet incorporated this work into either the SAM-T2K multiple-alignment procedure or the SAM-T02 multi-track HMMs for fold recognition.

D.2 New-fold techniques

Much of our effort will be dedicated to improving our techniques for handling proteins, protein complexes, and parts of proteins not covered by fold-recognition techniques.

D.2.1 Multimeric proteins

The conformation representation used by undertaker already handles non-contiguous backbones, and so is fairly easily extended to handle multiple protein chains. We plan to try predicting structures for both homo-multimers and hetero-multimers. We plan to start with homo-multimers, since we can then make the assumption that all the copies of the protein have the same conformation, and the only extra degrees of freedom are the rigid transformations that map one copy to another. The operators that we have defined for moving subtrees in a single chain should be applicable to multimeric proteins with only minor modifications.

D.2.2 Incorporating experimental data into predictions

Various experimental techniques (for example, NMR and cross-linking experiments) provide data about approximate distances between selected atoms. Undertaker accepts distance constraints expressed as a desired distance and a range around the desired distance that is used for shaping

the energy well around the minimum at the desired distance. Undertaker can use this information both in the cost function and in the conformation generation. We are interested in seeing how much of such data is needed to get reasonable structures, and how much noise can be tolerated. We have used hand-generated constraints to guide assembly of beta sheets, but have not yet tried using experimental data.

Recently, Carol Rohl has used Rosetta together with residual diploar couplings to predict protein backbones accurately with much less data than is usually needed for NMR determination of structure [102]. We have already included distance constraints and predicted torsion angle cost functions in undertaker, so modifying it further to handle the information from residual dipolar couplings should be straightforward. Although Carol Rohl is not a co-PI on this proposal, she has accepted an offer to be on the engineering faculty at University of California, Santa Cruz and will be collaborating with us.

D.2.3 Improving the cost function

One of the biggest areas for improvement in undertaker is the cost function that is used to determine whether or not a conformation looks like a real protein.

While trying to use undertaker for CASP5 predictions, the most glaring omission was the lack of a hydrogen-bond cost function. Undertaker tended to pull beta sheets apart, unless the conformation was very compact, in order to reduce breaks or clashes. We had to add constraints by hand for the hydrogen bonds we wanted to form or keep, in order to get reasonable results from undertaker. One of our highest priorities is to add hydrogen bonds to the cost function, and hydrogen bond optimization to the set of operators.

There are many different expressions that have been used for H-bond energy functions, which seem to disagree on what terms are most important. We will probably have to write our own cost function, based on histograms of measurements from high-resolution X-ray structures. Our representation, which uses all heavy-atoms, has an advantage over sidechain-centroid representations, in that sidechain hydrogen bonds can be detected and evaluated. Our lack of explicit hydrogens may be a problem, as we have to compute the cost of the hydrogen bond from the positions of just the donor, the acceptor, and the non-hydrogen atoms they are covalently bound to. For backbone hydrogen bonds, which are the most important ones to model, the lack of explicit hydrogens does not present any problems, since the position of the hydrogen on the backbone nitrogen is easily determined from the available atom positions.

In addition to the H-bond cost function(s), we will add more basic cost functions for predicted local properties—particularly those that have high predictability in our tests of local structure alphabets and are quickly measured in undertaker. We have developed a way to convert the discrete ALPHA alphabet predictions into a continuous density function for the underlying α torsion angle, but have not yet done experiments to optimize the parameters of this conversion. We have not yet developed a method for converting ANG alphabet predictions into density functions over (ϕ, ψ) pairs, but this is another obvious direction to try improving our application of predicted local structure to the cost function.

The cost function used in optimizing a structure is a linear combination of any number of different basic cost functions, but we have not yet determined a good weighting for the different terms. We generated a number of decoys for CASP5 that could be used set up a regression problem to optimize the weights, once the correct structures are released. We will also be setting up a benchmark set of

structures and decoys for testing new basic score functions and optimizing the weights of the score functions, since the CASP5 set is quite likely not a representative set of proteins.

We have found that existing decoy sets are not particularly useful for evaluating new cost functions [6], as they are most sensitive to terms that have been omitted from the cost function used when creating the decoys, and are very insensitive to terms included in that original cost function. This problem can be only partially addressed by new decoy test sets, so we will also have to evaluate any new cost function by doing optimizations using that cost function, not just scoring existing decoy sets.

D.2.4 Fragment libraries and conformation-change operators

In order for the fragment-packing technique to work, the fragment library it works from must contain fragments that are sufficiently close to the true structure. Although this can be guaranteed by using a large library of very short fragments, search is then too slow to be useful. Having a high density of low-RMSD medium-length and long fragments in the library greatly accelerates the conformational search.

Since undertaker uses alignments and fragments found with our HMM tools, we expect that improvements in the fold-recognition methods will also result in improved fragment libraries. We will add tests of fragment libraries to our protocol for testing local structure alphabets and multi-track HMMs, to find ways of generating better libraries.

At the moment, the medium-length and long fragments used by undertaker have had simple sidechain replacement done on them to tailor them for the particular position in the chain where they are needed. No attempt has been made to optimize the sidechains. Because of this, many of the fragments have internal clashes which make them score poorly when they are inserted. We plan to experiment with doing some sidechain rotamer optimization on the fragment libraries before beginning the main undertaker optimization of the whole conformation. We need to determine whether the expense of optimizing the fragments results in better optimization of the main chain, or whether we would be better off using cheaper fragment libraries, and spending the resources doing the sidechain optimization in the context of the whole chain, as we currently do.

In addition to having good pieces to work from, the search algorithm must be able make conformation changes that improve the structure, without getting trapped in local minima of the cost function. We have found that fragment-insertion alone gets trapped fairly easily and have been developing other operators to change the conformation.

One set of operators are *crossover* operators, which create a new conformation by splitting the chain into two parts, copying the first part from one conformation and the second part from a different conformation. We have found it useful to insert a fragment at the join, to prevent the pool of conformations from filling up with multiple copies of a single conformation. We plan to add more crossover operations (such as an ABA crossover, that has two breakpoints and copies the middle part from a different conformation, or a subtree replacement crossover), to allow easier combination of good features from different conformations.

We have also started creating operators that make directed changes, rather than random changes, to the conformation. For example, the OptSubtree operator splits off one part of the protein to reposition, and looks at all distance constraints, disulfide bonds, and peptide bonds that join the subtree to the rest of the protein. The subtree is then repositioned to try to minimize the distances between where it is and where the various bonds and constraints imply it should be. This operator drives down the cost function fairly quickly, but cannot be used alone, as it does only

rigid transformation of a subtree, and cannot reshape the backbone within the subtree. We plan to improve this operator to take into account clashes and hydrogen bonds also.

Because OptSubtree and the related JiggleSubtree, which moves the subtree randomly by a small amount, are so powerful, we are plannnig to add an operator that deliberately breaks the backbone where it is likely to be flexible (for example, at residues that are predicted to be on the surface and whose secondary structure predictions are weak), to allow more rapid movement of the rigid pieces, followed by fragment insertion in the neighborhood of the gap to close the break again.

Another operator we plan to add is one that tries to select fragments that will match strongly-predicted secondary structure, rather than selecting fragments randomly and checking the local structure properties only in the cost function. We can also encourage the rapid formation of local structure by automatically adding distance constraints for helix length or for residues on a strand that are separated by 2 or 4. Our longer fragments already are strongly biased in favor of the predicted structure, but the 1–4-residue-long generic fragments are not so selected.

When we have good alignments to templates, it would be useful to be able to concentrate undertaker's attention on the loops, so it does not waste so much of the time evaluating conformations that pull apart the core. Rosetta does this in an all-or-nothing way, by freezing the backbone in the core for most of the optimization. Although we might introduce the option of freezing the core, we are more interested in exploring other approaches, for example: increasing the probability of fragment insertion near breaks, particularly if we introduce breaks in flexible regions; applying two fragments simultaneously to residues that are close in 3-space but not on the chain, to get possible hinging motions; or doing fragment insertions that move gaps out of strongly-predicted regions.

When we have a compact homology model built, there are often small breaks and clashes that we have not been able to remove by fragment insertion. It might be good to allow small changes to ϕ and ψ angles to close these small breaks. One operation we will look at would adjust the torsion angles of a pair of backbone bonds that are nearly collinear, resulting in a movement of region between the bonds, without changing what is outside them. We are interested both in making small random changes and in trying to make changes that optimize parts of the cost function.

D.2.5 Human interface

It is easy to view the output of an undertaker optimization run with standard structure visualization tools such as Rasmol [108], but there is currently no interactive way to manipulate the conformation. There were many times during the CASP5 prediction season when we wanted to push a helix into the "obvious" place, or move a strand to get better sheet formation, but we had no easy way to do this. We were reduced to writing distance constraints to express where we wanted things to be and running perl scripts that moved some part of the conformation to a different location.

We do not want to get into the endless morass of graphical user interfaces, but do plan on adding some operations to make hand-manipulation of conformations easier. Undertaker can already be run interactively, and if we add an option that allows RasMol to display the current conformation without leaving undertaker, we should be able to make hand-tweaking of conformations fairly straightforward. In addition to direct hand-tweaking, we may want to apply specific undertaker operators or optimization steps, both for debugging the operators and for doing hand-directed changes to a conformation.

D.3 Web pages and outreach

We propose putting together a turnkey solution for biologists that would accept a target protein sequence and provide at least

1. a multiple alignment of probably homologous protein sequences from a protein data base such as NR [94],
2. local structure predictions (such as secondary structure) for each position in the chain, together with estimates of the reliability of the predictions, and an explanation of the alphabets being predicted,
3. sequence logos, showing graphically what positions in the multiple alignment have high conservation and where the local structure predictions are most reliable,
4. a short list of the most similar proteins currently in the Protein Data Bank [8], with calibrated E-values expressing how likely such a similarity is by chance,
5. high-quality alignments of the target sequence to the template proteins found in PDB,
6. simple three-dimensional models based on the alignments to templates,
7. complete three-dimensional models built by combining various alignments and fragments with undertaker to optimize an energy function.

All of these steps would also be provided as a free web service, though the last may need to be limited because of computational cost. Our current SAM-T02 web server[5] provides up to Step 5, but we plan to make significant improvements in all the steps.

We intend to increase the usage of our web server for protein-structure prediction by an order of magnitude in the next three years. We have seen a growth from 14,000 uses a year to 22,000 then 33,000 over the last three years. Continuing this growth rate would take five years to get an order of magnitude increase in use. To accelerate the usage of our tools, it is not enough to have the best predictions—we must also have a clear, easy-to-use site with good tutorials on structure prediction and documentation for interpreting the various outputs we provide, and we must inform people of the existence and value of our tools.

We plan to redesign our site for easier maintenance and expansion. Part of this redesign effort will be focussed on making the site easier to use and linking documentation in wherever possible. We will have to write most of this documentation from scratch, since existing research papers are generally too detailed and too technical for the intended end users, who are students and researchers in biology, not bioinformatics.

Outreach is a more difficult problem for us, as our intended users are scattered all over the world in different branches of biology. There is no small set of conferences, journal pages, web sites, or newsgroups that would reach more than a very small fraction of the audience.

One way we may be able to get more good predictions into the hands of bench biologists is to make predictions for the globular domains in the human genome, linking to the predictions from the human genome browser at genome.ucsc.edu. These predictions would be made in the same way as target proteins submitted over the web page, and the same outputs would be provided, but pre-computation would make them much more quickly available, and the links from the browser would make them easily accessible to a very large group of biologists and biomedical researchers. The developers of the browser have expressed serious interest in such a collaboration.

[5] http://www.soe.ucsc.edu/research/compbio/HMM-apps/T02-query.html

G Literature Cited

[1] S. Altschul, W. Gish, W. Miller, E. Myers, and D. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410, 1990.

[2] S. Altschul, T. Madden, A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. Lipman. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Research*, 25:3899–3402, 1997.

[3] Stephen F. Altschul. Amino acid substitution matrices from an information theoretic perspective. *Journal of Molecular Biology*, 219:555–565, 1991.

[4] K. Asai, S. Hayamizu, and K. Onizuka. HMM with protein structure grammar. In *Proceedings of the Hawaii International Conference on System Sciences*, pages 783–791, Los Alamitos, CA, 1993. IEEE Computer Society Press.

[5] P. Baldi, Y. Chauvin, T. Hunkapillar, and M. McClure. Hidden Markov models of biological primary sequence information. *Proceedings of the National Academy of Sciences, USA*, 91:1059–1063, 1994.

[6] Christian Barrett. *Investigation of Non-pairwise Protein Structure Score Functions Using Sets of Decoy Structures*. PhD thesis, University of California, Computer Science, UC Santa Cruz, CA 95064, 2001.

[7] Christian Barrett, Richard Hughey, and Kevin Karplus. Scoring hidden Markov models. *Computer Applications in the Biosciences*, 13(2):191–199, 1997.

[8] F.C. Bernstein, T. F. Koetzle, G. J. Williams, E. E. Meyer, M. D. Brice, J. R. Rodgers, O. Kennard, T. Shimanouchi, and M. Tasumi. The Protein Data Bank: a computer-based archival file for macromolecular structures. *Journal of Molecular Biology*, 112:535–542, 1977.

[9] Richard Bonneau, Jerry Tsai, Ingo Ruczinski, Dylan Chivian, Carol Rohl, Charlie E. M. Strauss, and David Baker. Rosetta in CASP4: progress in ab initio protein structure prediction. *Proteins: Structure, Function, and Genetics*, 45(S5):119–126, 2001.

[10] J. U. Bowie, R. Lüthy, and D. Eisenberg. A method to identify protein sequences that fold into a known three-dimensional structure. *Science*, 253:164–170, 1991.

[11] Philipp Bucher, Kevin Karplus, Nicolas Moeri, and Kay Hoffman. A flexible motif search technique based on generalized profiles. *Computers and Chemistry*, 20(1):3–24, January 1996.

[12] Janusz M. Bujnicki, Arne Elofsson, Daniel Fischer, and Leszek Rychlewski. Livebench-2: Large-scale automated evaluation of protein structure prediction servers. *Proteins: Structure, Function, and Genetics*, 45(S5):184–191, 2001.

[13] C. Burge and S. Karlin. Predictions of complete gene structures in human genomic DNA. *Journal of Molecular Biology*, 268:78–94, 1997.

[14] C. Bystroff and D. Baker. Prediction of local structure in proteins using a library of sequence-structure motifs. *Journal of Molecular Biology*, 281:565–577, 1998.

[15] C. Bystroff, V. Thorsson, and D. Baker. Hmmstr: a hidden markov model for local sequence-structure co rrelations in proteins. *Journal of Molecular Biology*, 301(1):173–190, Aug 2000.

[16] A.C. Camproux, P. Tuffery, J.P. Chevrolat, J.F. Boisvieux, and S. Hazout. Hidden markov model approach for identifying the modular framework of the protein backbone. *Protein Eng.*, 12(12):1063–1073, Dec 1999.

[17] L. R. Cardon and G. D. Stormo. Expectation maximization algorithm for identifying protein-binding sites with variable lengths from unaligned DNA fragments. *Journal of Molecular Biology*, 223:159–170, 1992.

[18] G. A. Churchill. Stochastic models for heterogeneous DNA sequences. *Bull Math Biol*, 51:79–94, 1989.

[19] Melissa Cline, Richard Hughey, and Kevin Karplus. Predicting reliable regions in protein sequence alignments. *Bioinformatics*, 18:306–324, 2002.

[20] Melissa Cline and Kevin Karplus. On alignment shift and its measures. Technical Report UCSC-CRL-97-27, University of California, Santa Cruz, Jack Baskin School of Engineering, UC Santa Cruz, CA 95064, February 1998.

[21] Melissa S. Cline, Kevin Karplus, Richard H. Lathrop, Temple F. Smith, Robert G. Rogers Jr., and David Haussler. Information-theoretic dissection of pairwise contact potentials. *Proteins: Structure, Function, and Genetics*, 49(1):7–14, 1 October 2002.

[22] X. De La Cruz and J.M. Thornton. Factors limiting the performance of prediction-based fold recognition methods. *Protein Science*, 8:750–759, 1999.

[23] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt. A model of evolutionary change in proteins. In *Atlas of Protein Sequence and Structure*, chapter 22, pages 345–358. National Biomedical Research Foundation, Washington, D. C., 1978.

[24] A.G. de Brevern, C. Etchebest, and S. Hazout. Bayesian probabilistic approach for predicting backbone struc tures in terms of protein blocks. *Proteins: Structure, Function and Genetics*, 41:271–287, 2000.

[25] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.

[26] Sean Eddy. Multiple alignment using hidden Markov models. In Christopher Rallings et al., editors, *Proceedings, 3rd International Conference on Intelligent Systems for Molecular Biology*, pages 114–120, Menlo Park, CA, July 1995. AAAI/MIT Press.

[27] S.R. Eddy, G. Mitchison, and R. Durbin. Maximum discrimination hidden Markov models of sequence consensus. *J. Comput. Biol.*, 2:9–23, 1995.

[28] Robert Edgar. Profile-profile alignment using hidden Markov models. unpublished, 2002.

[29] VA Eyrich, MA Marti-Renom, D Przybylski, MS Madhusudhan, A Fiser, F Pazos, A Valencia, A Sali, and B Rost. EVA: continuous automatic evaluation of protein structure prediction servers. *Bioinformatics*, 17(12):1242–1243, December 2001.

[30] J.S. Fetrow, M.J. Palumbo, and G. Berg. Patterns, structures, and amino acid frequencies in structural building blocks, a protein secondary structure classification scheme. *Proteins: Structure, Function, and Genetics*, 27:249–271, 1997.

[31] V. Di Francesco, J. Garnier, and P.J. Munson. Protein topology recognition from secondary structure sequences: application of the hidden Markov models to the alpha class proteins. *Journal of Molecular Biology*, 267(2):446–463, 1997.

[32] V. Di Francesco, V. Geetha, J. Garnier, and P.J. Munson. Fold recognition using predicted secondary structure sequences and hidden markov models of protein folds. *Proteins: Structure, Function and Genetics, Suppl.*, 1:123–128, 1997.

[33] Dimitrij Frishman and Patrick Argos. Knowledge-based protein secondary structure assignment. *Proteins: Structure, Function, and Genetics*, 23:566–579, 1995.

[34] O. Gotoh. Multiple sequence alignment: algorithms and applications. *Advances in Biophysics*, 36(1):159–206, 1999.

[35] Julian Gough, Kevin Karplus, Richard Hughey, and Cyrus Chothia. Assignment of homology to genome sequences using a library of hidden Markov models that represent all proteins of known structure. *Journal of Molecular Biology*, 313(4):903–919, 2 Nov 2001.

[36] M. Gribskov, R. Lüthy, and D. Eisenberg. Profile analysis. *Methods in Enzymology*, 183:146–159, 1990.

[37] M. Gribskov and N.L. Robinson. Use of receiver operating characteristic (roc) analysis to evaluate sequence matching. *Computers Chem.*, 20(1):25–33, 1996.

[38] Michael Gribskov, Andrew D. McLachlan, and David Eisenberg. Profile analysis: Detection of distantly related proteins. *Proceedings of the National Academy of Sciences, USA*, 84:4355–4358, July 1987.

[39] S. Griffiths-Jones and A. Bateman. The use of structure information to increase alignment accuracy does not aid homlogue detection with profile hmms. *Bioinformatics*, 18(9):1243–1249, 2002.

[40] Genetics Computer Group. *Program Manual for the GCG Package, Version 7.* Genetics Computer Group, 575 Science Drive, Madison, Wisconsin, USA 53711, April 1991.

[41] W. N. Grundy, W. Bailey, T. Elkan, and C. Baker. Meta-MEME: Motif-based hidden Markov models of protein families. *Computer Applications in the Biosciences*, 13(4):397–406, 1997.

[42] S. S. Hannenhalli and R. B. Russell. Analysis and prediction of functional sub-types from protein sequence alignments. *Journal of Molecular Biology*, 303:62–76, 2000.

[43] J. Hargbo and A. Elofsson. Hidden markov models that use predicted secondary structures for fold recognition. *Proteins: Structure, Function, and Genetics*, 36:68–76, 1999.

[44] D. Haussler, A. Krogh, I. S. Mian, and K. Sjölander. Protein modeling using hidden Markov models: Analysis of globins. In *Proceedings of the Hawaii International Conference on System Sciences*, volume 1, pages 792–802, Los Alamitos, CA, 1993. IEEE Computer Society Press.

[45] Steven Henikoff and Jorja G. Henikoff. Automated assembly of protein blocks for database searching. *Nucleic Acids Research*, 19(23):6565–6572, 1991.

[46] D. G. Higgins, J. D. Thompson, and T. J. Gibson. Using CLUSTAL for multiple sequence alignments. *Methods in Enzymology*, 266:383–402, 1996.

[47] L. Holm and C. Sander. Protein folds and families: sequence and structure alignments. *Nucleic Acids Research*, 27:244–247, 1999.

[48] Liisa Holm and Chris Sander. Protein structure comparison by alignment of distance matrices. *Journal of Molecular Biology*, 233(1):123–138, 5 Sept 1993.

[49] Liisa Holm and Chris Sander. Mapping the protein universe. *Science*, 273(5275):595–603, Aug 2 1996.

[50] T. Hubbard, A. Murzin, S. Brenner, and C. Chothia. SCOP: a structural classification of proteins database. *Nucleic Acids Research*, 25(1):236–9, January 1997.

[51] Tim J.P. Hubbard. Fold recognition and ab initio structure predictions using hidden Markov models and beta-strand pair potentials. *Proteins: Structure, Function, and Genetics*, 23(3):398–402, November 1995.

[52] R. Hughey and A. Krogh. SAM: Sequence alignment and modeling software system. Technical Report UCSC-CRL-95-7, University of California, Santa Cruz, Computer Engineering, UC Santa Cruz, CA 95064, 1995.

[53] Richard Hughey, Kevin Karplus, and Anders Krogh. SAM: Sequence alignment and modeling software system, version 3. Technical Report UCSC-CRL-99-11, University of California, Santa Cruz, Computer Engineering, UC Santa Cruz, CA 95064, October 1999. Available from `http://www.soe.ucsc.edu/research/compbio/sam.html`.

[54] Richard Hughey and Anders Krogh. Hidden Markov models for sequence analysis: Extension and analysis of the basic method. *Computer Applications in the Biosciences*, 12(2):95–107, 1996. Information on obtaining SAM is available at `http://www.soe.ucsc.edu/research/compbio/sam.html`.

[55] BioInfoBank Institute. Livebench, ongoing experiment. http://bioinfo.pl/LiveBench/.

[56] K. Fasman J. Henderson, S. Salzberg. Finding genes in DNA with a hidden Markov model. *Journal of Computational Biology*, 4:127–141, 1997.

[57] L. Jaroszewski, L. Rychlewski, and A. Godzik. Improving the quality of twilight-zone alignments. *Protein Science*, 9:1487–1496, 2000.

[58] F. Jeanmougin, J. Thompson, M. Gouy, D. Higgins, and T. Gibson. Multiple sequence alignment with Clustal X. *Trends in Biochemical Sciences*, 23(10):403–5, 1998.

[59] D.T. Jones. Protein secondary structure prediction based on position-specific scoring matrices. *Journal of Molecular Biology*, 292:195–202, 1999.

[60] W. Kabsch and C. Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22(12):2577–2637, Dec 1983.

[61] H.S. Kang, N.A. Kurochkina, and B. Lee. Estimation and use of protein backbone angle probabilities. *Journal of Molecular Biology*, 229:448–460, 1993.

[62] R. Karchin, M. Cline, Y. Mandel-Gutfreund, and K. Karplus. Hidden Markov models that use predicted local structure for fold recognition: alphabets of backbone geometry. unpublished, 2002.

[63] R. Karchin and R. Hughey. Weighting hidden Markov models for maximum discrimination. *Bioinformatics*, 14(9):772–782, 1998.

[64] Rachel Karchin. Classifying g-protein coupled receptors with support vector machines. Master's thesis, University of California, Computer Science, UC Santa Cruz, CA 95064, 2000.

[65] K. Karplus, R. Karchin, and R. Hughey. Calibrating E-values for hidden Markov models with reverse-sequence null models. Unpublished, 2002.

[66] Kevin Karplus. Predicting protein structure using SAM, UCSC's hidden Markov model tools. In Igor F. Tsigelny, editor, *Protein Structure Prediction: Bioinformatic Approach*, IUL Biotechnology Series, pages 297–323. International University Line, La Jolla, California, 2002.

[67] Kevin Karplus, Christian Barrett, Melissa Cline, Mark Diekhans, Leslie Grate, and Richard Hughey. Predicting protein structure using only sequence information. *Proteins: Structure, Function, and Genetics*, Supplement 3(1):121–125, 1999.

[68] Kevin Karplus, Christian Barrett, and Richard Hughey. Hidden Markov models for detecting remote protein homologies. *Bioinformatics*, 14(10):846–856, 1998.

[69] Kevin Karplus and Birong Hu. Evaluation of protein multiple alignments by SAM-T99 using the BaliBASE multiple alignment test set. *Bioinformatics*, 17:713–720, August 2001.

[70] Kevin Karplus, Rachel Karchin, Christian Barrett, Spencer Tu, Melissa Cline, Mark Diekhans, Leslie Grate, Jonathan Casper, and Richard Hughey. What is the value added by human intervention in protein structure prediction? *Proteins: Structure, Function, and Genetics*, 45(S5):86–91, 2001.

[71] Kevin Karplus, Kimmen Sjölander, Christian Barrett, Melissa Cline, David Haussler, Richard Hughey, Liisa Holm, and Chris Sander. Predicting protein structure using hidden Markov models. *Proteins: Structure, Function, and Genetics*, Suppl. 1:134–139, 1997.

[72] L.A. Kelley, R.M. MacCallum, and M.J.E. Sternberg. Enhanced genome annotation using structural profiles in the program 3d-pssm. *Journal of Molecular Biology*, 299:501–522, 2000.

[73] S.M. King and W.C. Johnson. Assigning secondary structure from protein coordinate data. *Proteins: Structure, Function, and Genetics*, 35:313–320, 1999.

[74] T. Kohonen. Self-organizing formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69, 1982.

[75] T. Kohonen. *Self-organization and associative memory*. Springer-Verlag, third edition, 1989.

[76] A. Krogh, M. Brown, I. S. Mian, K. Sjölander, and D. Haussler. Hidden Markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235:1501–1531, February 1994.

[77] A. Krogh, I. S. Mian, and D. Haussler. A Hidden Markov Model that finds genes in *E. coli* DNA. *Nucleic Acids Research*, 22:4768–4778, 1994.

[78] D. Kulp, D. Haussler, M.G. Reese, and F. Eeckman. A generalized hidden Markov model for the recognition of human genes in DNA. In *Proceedings, 4rd International Conference on Intelligent Systems for Molecular Biology*, pages 134–142, St. Louis, June 1996. AAAI Press. http://www.soe.ucsc.edu/~dkulp/cgi-bin/genie.

[79] E. S. Lander and P. Green. Construction of multilocus genetic linkage maps in humans. *Proceedings of the National Academy of Sciences, USA*, 84:2363–2367, 1987.

[80] C. E. Lawrence, S. Altschul, M. Boguski, J. Liu, A. Neuwald, and J. Wootton. Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment. *Science*, 262:208–214, 1993.

[81] C. E. Lawrence and A. A. Reilly. An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *Proteins: Structure, Function, and Genetics*, 7:41–51, 1990.

[82] Lawrence Livermore National Laboratory. CASP5 experiment web site. http://predictioncenter.llnl.gov/casp5/Casp5.html, 2002.

[83] Arthur M. Lesk, Loredana Lo Conte, and Tim J. P. Hubbard. Assessment of novel fold targets in CASP4: Predictions of three-dimensional structures, secondary structures, and interresidue contacts. *Proteins: Structure, Function, and Genetics*, 45(S5):98–118, 2001.

[84] J. M. Levin, S. Pascarella, P. Argos, and J. Garnier. Quantification of secondary structure prediction improvement using multiple alignments. *Protein Engineering*, 6:849–854, 1993.

[85] Michael Levitt. Competetive assessment of protein fold recognition and alignment accuracy. *Proteins: Structure, Function, and Genetics*, Supplement 1(1):92–104, 1997.

[86] J. MacQueen. Some methods for classification and analysis of multivariate observations. In L.M. Le Cam and J. Neyman, editors, *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. University of California Press, Berkeley, 1967.

[87] Martin Madera and Julian Gough. A comparison of profile hidden Markov model procedures for remote homology detection. *Nucleic Acids Research*, 30(19):in press, 2002.

[88] Marcella McClure, Chris Smith, and Pete Elton. Parameterization studies for the SAM and HMMER methods of hidden Markov model generation. In *Proceedings, 4rd International Conference on Intelligent Systems for Molecular Biology*, pages 155–164, St. Louis, June 1996. AAAI Press.

[89] B. Morgenstern, K. French, A. Dress, and T. Werner. Dialign: finding local similarities by multiple sequence alignment. *Bioinformatics*, 14(3):290–4, 1998.

[90] J. Moult, T. Hubbard, K. Fidelis, and J. Pedersen. Critical assessment of methods of protein structure prediction (CASP): round III. *Proteins: Structure, Function, and Genetics*, Supplement 3(1):2–6, 1999.

[91] Alexey G. Murzin. Structure classification-based assessment of CASP3 predictions for the fold recognition targets. *Proteins: Structure, Function, and Genetics*, Supplement 3(1):88–103, 1999.

[92] Cedric Notredame and Desmond G. Higgins. SAGA: sequence alignment by genetic algorithm. *Nucleic Acids Research*, 24:1515–1524, 1996.

[93] Cedric Notredame, Desmond G. Higgins, and Jaap Heringa. T-Coffee: A novel method for fast and accurate multiple sequence alignment. *Journal of Molecular Biology*, 302:205–217, 2000.

[94] NR (All non-redundant GenBank CDS translations+PDB+SwissProt+PIR+PRF Database) Distributed on the Internet via anonymous FTP from `ftp://ftp.ncbi.nlm.nih.gov/blast/db`. Information on NR is available at `http://www.ncbi.nlm.nih.gov/BLAST/blast_databases.html`.

[95] C.A. Orengo, J.E. Bray, T. Hubbard, L. LoConte, and I. Sillitoe. Analysis and assessment of ab initio three-dimensional prediction, secondary structure, and contacts prediction. *Proteins: Structure, Function, and Genetics*, Supplement 3(1):149–170, 1999.

[96] J. Park, K. Karplus, C. Barrett, R. Hughey, D. Haussler, T. Hubbard, and C. Chothia. Sequence comparisons using multiple sequences detect three times as many remote homologues as pairwise methods. *Journal of Molecular Biology*, 284(4):1201–1210, 1998. Paper available at `http://www.mrc-lmb.cam.ac.uk/genomes/jong/assess_paper/assess_paperNov.html`.

[97] S. Pascarella, R. De Persio, F. Bossa, and P. Argos. Easy method to predict solvent accessibility from multiple protein sequence alignments. *Proteins: Structure, Function, and Genetics*, 32:190–199, 1998.

[98] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[99] G.N. Ramachandran, C. Ramakrishnan, and V. Sasisekharan. Stereochemistry of polypeptide chain configurations. *Journal of Molecular Biology*, 7:95–99, 1963.

[100] M. G. Reese, F. H. Eeckman, D. Kulp, and D. Haussler. Improved splice site detection in genie. *Journal of Computational Biology*, 4:311–323, 1997.

[101] D.W. Rice and D. Eisenberg. A 3d-1d substitution matrix for protein fold recognition that includes predicted secondary structure of the sequence. *Journal of Molecular Biology*, 267(4):1026–1038, 1997.

[102] Carol Rohl and David Baker. De novo determination of protein backbone structure from residual dipolar couplings using Rosetta. *Journal of the American Chemical Society*, 124(11):2723–2729, 20 March 2002.

[103] M.J. Rooman, J. Rodriguez, and S.J. Wodak. Automatic definition of recurrent local structure motifs in proteins. *Journal of Molecular Biology*, 213:327–336, 1990.

[104] M.J. Rooman, J. Rodriguez, and S.J. Wodak. Relations between protein sequence and structure and their significance. *Journal of Molecular Biology*, 213:337–350, 1990.

[105] B. Rost. Phd: predicting one-dimensional protein structure by profile-based neural networks. *Methods in Enzymology*, 2666:525–39, 1996.

[106] B. Rost, C. Sander, and R. Schneider. Redefining the goals of protein secondary structure prediction. *Journal of Molecular Biology*, 235:13–26, 1994.

[107] B. Rost, R. Schneider, and C. Sander. Protein fold recognition by prediction-based threading. *Journal of Molecular Biology*, 270:471–480, 1997.

[108] Roger Sayle and E. James Milner-White. RasMol: Biomolecular graphics for all. *Trends in Biochemical Sciences*, 20(9):374–376, September 1995.

[109] Alejandro A. Schäffer, L. Aravind, Thomas L. Madden, Sergei Shavirin, John L. Spouge, Yuri I. Wolf, Eugene Koonin, and Stephen F. Altschul. Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements. *Nucleic Acids Research*, 29(14):2994–3005, 2001.

[110] I. N. Shindyalov and P. E. Bourne. Protein structure alignment by incremental combinatorial extension (ce) of the optimal path. *Protein Engineering*, 11(9):739–47, 1998.

[111] Kim T. Simons, Rich Bonneau, Ingo Ruczinski, and David Baker. Ab initio protein structure prediction of CASP III targets using ROSETTA. *Proteins: Structure, Function, and Genetics*, Supplement 3(1):171–176, 1999.

[112] Manfred J. Sippl, Peter Lackner, Francisco S. Domingues, Andreas Prlić, Rainer Maik, Antonina Andreeva, and Markus Wiederstein. Assessment of the CASP4 fold recognition category. *Proteins: Structure, Function, and Genetics*, 45(S5):55–67, 2001.

[113] K. Sjölander, K. Karplus, M. P. Brown, R. Hughey, A. Krogh, I. S. Mian, and D. Haussler. Dirichlet mixtures: A method for improving detection of weak but significant protein sequence homology. *Computer Applications in the Biosciences*, 12(4):327–345, August 1996.

[114] R. F. Smith and T. F. Smith. Pattern-induced multi-sequence alignment (PIMA) algorithm employing secondary structure-dependent gap penalties for use in comparative protein modelling. *Protein Engineering*, 5(1):35–41, 1992.

[115] T. F. Smith and M. S. Waterman. Comparison of bio-sequences. *Advances in Applied Mathematics*, 2:482–489, 1981.

[116] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[117] C. M. Stultz, J. V. White, and T. F. Smith. Structural analysis based on state-space modeling. *Protein Science*, 2:305–315, 1993.

[118] M.B. Swindells, M.W. MacArthur, and J.M. Thornton. Intrinsic phi,psi propensities of amino acids, derived from the coil regions of known structures. *Nat. Struct. Biol.*, 2(7):596–603, Jul 1995.

[119] Christoper Tarnas and Richard Hughey. Reduced space hidden Markov model training. *Bioinformatics*, 14(5):401–406, 1998.

[120] Julie D. Thompson, Desmond G. Higgins, and Toby J. Gibson. CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties, and weight matrix choice. *Nucleic Acids Research*, 22(22):4673–4680, 1994.

[121] M.J. Thompson and R.A. Goldstein. Predicting protein secondary structure with probabilistic schemata of evolutionarily derived information. *Protein Science*, 6:1963–1975, 1997.

[122] R. Unger, D. Harel, Wherland S., and J. Sussman. A 3d building blocks approach to analyzing and predicting structure of proteins. *Proteins*, 5:355–373, 1989.

[123] R. Unger and J.L Sussman. The importance of short structural motifs in protein structure analysis. *J Comput Aided Mol Des*, 7(4):457–472, 1993.

[124] A. J. Viterbi. Error bounds for convolution codes and an asymptotically optimal decoding algorithm. *IEEE Trans. Informat. Theory*, 13:260–269, 1967.

[125] D. H. Wolpert and D. R. Wolf. Estimating functions of probability distributions from a finite set of samples; part 2: Bayes estimators for mutual information, chi-squared, covariance, and other statistics. Technical Report LA-UR-93-833,TR-93-07-047, Los Alamos National Lab, Santa Fe Institute, 1993.

[126] D. H. Wolpert and D. R. Wolf. Estimating functions of probability distributions from a finite set of samples. *Physical Review E*, 52(6):6841–54, December 1995.

[127] T. Yada, T. Sazuka, and M. Hirosawa. Analysis of sequence patterns surrounding the translation initiation sites on cyanobacterium genome using the hidden Markov model. *DNA Research*, 4:1–7, 1997.

[128] Golan Yona and Michael Levitt. Within the twilight zone: a sensitive profile-profile comparison tool based on information theory. *Journal of Molecular Biology*, 315:1257–1275, 2002.

[129] X. Zhang, J.S. Fetrow, W.A. Rennie, D.L. Waltz, and G. Berg. Automatic derivation of substructures yields novel structural building blocks in globular proteins. *Proceedings, 1st International Conference on Intelligent Systems for Molecular Biology*, pages 438–446, 1993.