

Running title: Covariance models of RNA

RNA Sequence Analysis Using Covariance Models

Sean R. Eddy and Richard Durbin
MRC Laboratory of Molecular Biology
Hills Road
Cambridge CB2 2QH
England

sre@mrc-lmb.cam.ac.uk, rd@mrc-lmb.cam.ac.uk

April 28, 1994

Abstract

We describe a general approach to several RNA sequence analysis problems using probabilistic models that flexibly describe the secondary structure and primary sequence consensus of an RNA sequence family. We call these models “covariance models”. A covariance model of tRNA sequences is an extremely sensitive and discriminative tool for searching for additional tRNAs and tRNA-related sequences in sequence databases. A model can be built automatically from an existing sequence alignment. We also describe an algorithm for learning a model and hence a consensus secondary structure from initially unaligned example sequences and no prior structural information. Models trained on unaligned tRNA examples correctly predict tRNA secondary structure and produce high-quality multiple alignments. The approach may be applied to any family of small RNA sequences.

Subject: Computational biology

Keywords: tRNA, covariance model, stochastic context-free grammar, database searching, multiple alignment, RNA secondary structure

Introduction

A major role of computational methods in molecular biology is to identify similarities between sequences. Similarity between sequences generally implies functional and/or evolutionary homology and therefore provides important biological information. The analysis of large-scale genome sequence data is particularly dependent upon similarity searching methods [1, 2, 3, 4]. Similarity searching methods are fairly well developed for protein sequence analysis. Fast algorithms such as BLAST [5] and FASTA [6] are in widespread use for detecting homologues of new protein sequences. Even more sensitive methods such as profiles [7, 8] or hidden Markov models [9, 10] are available which use consensus information from multiple sequence alignments to detect new members of protein sequence families.

There are also many biologically important macromolecules that are composed of RNA. These include transfer RNA [11, 12], ribosomal RNA [13], group I and group II catalytic introns [14, 15], and spliceosomal small nuclear RNAs [16], to name just a few. Target sites for genetic regulation are often specific structures in mRNA molecules, such as the TAR or RRE binding sites in the human immunodeficiency virus genome [17] or the iron response elements in ferritin and transferrin receptor mRNA [18]. *In vitro* selection methods select families of small RNA molecules fit for a particular function, such as protein binding [19, 20] or even catalysis [21], out of randomized repertoires. One wants to be able to detect similar RNAs and RNA motifs in sequence data. However, the primary sequence based techniques that generally work quite well for protein sequence analysis are not well suited for studying RNA.

Most functional RNAs appear to be selected more for maintenance of a particular base-paired structure than conservation of primary sequence. RNA secondary structure induces strong pairwise correlations in RNA sequence, usually manifested as Watson-Crick complementarity. RNA sequence analysis therefore must work with this pattern of correlations in addition to primary sequence conservation, and methods for searching databases for new members of RNA families have consequently lagged behind those for analysis of protein. Transfer RNA or group I introns can be recognized by specialized, custom-built programs [22, 23, 24, 25]. Programs that use manually constructed and relatively inflexible patterns of conserved residues and base-pairs, analogous to PROSITE patterns of protein motif sequences [26], have been described for RNA [27, 28]. More general methods that capture both primary and secondary structure consensus information while still flexibly scoring insertions, deletions, and mismatches are desirable [29, 30].

Database searching for RNAs is not the only problem affected by the lack of mathe-

mational models that deal with secondary structure. Multiple RNA sequence alignment, a prerequisite for the inference of phylogenetic trees and for RNA structure prediction, is a markedly circular problem: accurate multiple alignment relies on an accurate secondary structure prediction, and *vice versa*. RNA sequences that share a common function and structure can appear to be unrelated and unalignable until a common secondary structure is recognized. The most reliable means of consensus RNA secondary structure prediction and multiple alignment is the iterative, laborious refinement process of comparative sequence analysis [31, 32] – a process of computer-aided recognition of strongly correlated positions in a multiple alignment followed by manual refinement of the alignment. The rapid discovery of new RNA sequence families by *in vitro* selection methods, in particular, is creating a need for automatic RNA structure prediction and multiple alignment methods [21, 19, 33, 20].

Here we introduce a probabilistic model, which we call a “covariance model” (CM), which cleanly describes both the secondary structure and the primary sequence consensus of an RNA. Using covariance models, we introduce new and general approaches to several RNA analysis problems: consensus secondary structure prediction, multiple sequence alignment, and database similarity searching. We describe a dynamic programming algorithm for efficiently finding the globally optimal alignments of RNA sequences to a model, and we show how to use this algorithm for database searching. Covariance models are constructed automatically from existing RNA sequence alignments or even from initially unaligned example sequences, using an iterative training procedure that is essentially an automatic implementation of comparative sequence analysis and an algorithm that we believe is the first optimal algorithm for RNA secondary structure prediction based on pairwise covariations in multiple alignments.

We test these algorithms using data taken from a trusted alignment of 1415 tRNA sequences [12], and on genomic sequence data from the *C. elegans* genome sequencing project [4]. We find that an automatically constructed tRNA CM is significantly more sensitive for database searching than even the best custom-built tRNA searching program. Our methods produce tRNA alignments of higher accuracy than other automatic methods and they invariably predict the correct consensus cloverleaf tRNA secondary structure when given unaligned example tRNA sequences.

Methods

Description of a covariance model

An RNA covariance model is based on an ordered tree. A tree can capture all the pairwise interactions of an RNA secondary structure [34]. However, a tree cannot capture other tertiary structural interactions, such as non-pairwise interactions (base triples) or non-nested pairs (pseudoknots) [35]; the consequences of these approximations will be discussed later. Figure 1 shows an ordered tree representation of a single RNA sequence. Many different trees can represent any given sequence, but the “best” trees, for our purposes, will be those in which pairwise nodes of the tree are assigned to base pairs in the RNA structure and singlet nodes of the tree are assigned to single-stranded bases. Both the structure and the primary sequence are described by such a tree. The primary sequence can be regenerated (emitted) by a traversal of the tree from root to leaves and left to right.

This tree is an inflexible description of a single RNA. We need to allow for insertions, deletions, and mismatches to describe a family of related RNAs; therefore we imagine that each node describes columns in a multiple alignment instead of bases in an individual sequence. Specific base assignments are replaced with *symbol emission probabilities* assigned to the 16 possible pairwise nucleotide combinations or 4 singlet nucleotides. To accommodate minor variations in structure such as insertions and deletions, each node is replaced with a number of different *states* (Figure 2) with particular properties. “Match” states (MATP, MATL, MATR) account for the conserved consensus columns of an alignment. “Insert” states (INSL, INSR) account for insertions relative to the consensus. “Delete” states (DEL) emit nothing and allow for the possibility of deletions relative to the consensus. MATL and MATR singlet states are included in pairwise nodes to allow for the possibility that either side of a consensus base pair may be deleted to leave a bulge. States are connected to each other by *state transition probabilities*, which are scores for transitioning to one of several possible new states. For example, note that after entering an insert state, there is a state transition probability for staying in it; this roughly corresponds to a gap-open and gap-extend penalty. Special non-emitting states describe the tree structure itself, such as bifurcation states with forced transitions into two new states and dummy begin and end states. The state transition scores will favor a probable main line through the model that represents the consensus structure.

We call the resulting probabilistic model a covariance model. The final model consists of a number of states M , symbol emission probabilities \mathcal{P} , and state transition probabili-

ties \mathcal{T} . A CM can be thought of as a probabilistic machine that generates representative sequences of an RNA family. It describes an RNA multiple sequence alignment in terms of both primary sequence consensus and pairwise covariations induced by consensus secondary structure. CMs are a generalization of hidden Markov models (HMMs), probabilistically rigorous models which have been widely applied in speech recognition [36] and, more recently, applied to profile-like methods of protein sequence analysis [9, 10]. An HMM is a special case of a CM with no bifurcations and with no pairwise states for describing covariations. The algorithms for applying CMs to sequence analysis correspond to those for HMMs [36, 10], extended to tree structures.

Alignment algorithm

The basic operation when using CMs is the alignment of one RNA sequence to a CM and the calculation of a probability score. Multiple sequence alignments are produced by aligning individual sequences to a single model. A model is trained by optimizing the parameters and structure of the model to assign high alignment scores to a set of example training sequences. Secondary structures are predicted from what base pairs are assigned to pairwise states in an alignment. Database searching involves looking for high-scoring alignments to subsequences of arbitrarily long sequences.

The optimal alignment of an RNA sequence to a CM and its probability score are calculated using a three-dimensional dynamic programming algorithm. The key idea is to start from alignments of the smallest subtrees of the model to the smallest internal subsequences (single symbols and empty strings) and to use these smaller alignments to recursively calculate the probability for alignments of larger subtrees to larger subsequences until a globally optimum alignment has been calculated.

Specifically, a three-dimensional matrix is calculated, containing scores $S_{i,j,y}$ which are the log likelihoods of alignments of subsequences $i..j$ ($1 \leq i \leq j \leq N$ bases) to subtrees beginning in a state y ($1 \leq y \leq M$ states). We think of i and j as rows and columns, respectively, and y as levels. The states of the tree are numbered from the root, such that downstream (more terminal) states y_{next} always have higher indices than their parent state y ; i.e., in preorder traversal [37]. $\mathcal{T}(y_{next} | y)$ is a transition probability from state y to one possible downstream state y_{next} . $\mathcal{P}(x_i, x_j | y)$ is the probability that a state y emits the symbols x_i to the left and x_j to the right.

$S_{i,j,y}$ is found by calculating scores for each of up to six possible matrix cells that matrix cell i, j, y can connect to, and keeping the maximum score. Each possible $S_{i,j,y}$ is the sum

of three numbers: 1) the symbol emission log probability for i and/or j (or zero) depending on what kind of state y is; 2) the state transition log probability for moving from y to y_{next} ; and 3) the score $S_{i',j',y_{next}}$ for y_{next} aligned to a subsequence i',j' which is already known from previous steps in the recursion. Both i' and j' depend on what type of state y is; because y may emit x_i, x_j , both, or neither, depending on whether it is a singlet, pairwise, or non-emitting state, i' may be i or $i + 1$, and j' may be either j or $j - 1$.

The calculation begins by allocating and initializing a partial cube of $j = [0..N]$ rows, $i = [0..j + 1]$ columns, and $y = [1..M]$ levels. A set of “off-diagonal” scores $S_{j+1,j,y}$ handles boundary conditions which represent the score of the subtree from state y aligned to no sequence; the score of end states aligned to these is initialized to zero. All other scores are initialized to $-\infty$. Then, starting from the off-diagonal $i = j + 1, j$ and working towards the corner $i = 0, j = N$, we loop over $y = M$ down to $y = 1$ for each subsequence:

$$\begin{aligned}
 S_{i,j,y}(y = MATP) &= \max_{y_{next}} [S_{i+1,j-1,y_{next}} + \log \mathcal{T}(y_{next} | y) + \log \mathcal{P}(x_i, x_j | y)] \\
 S_{i,j,y}(y = MATL, INSL) &= \max_{y_{next}} [S_{i+1,j,y_{next}} + \log \mathcal{T}(y_{next} | y) + \log \mathcal{P}(x_i | y)] \\
 S_{i,j,y}(y = MATR, INSR) &= \max_{y_{next}} [S_{i,j-1,y_{next}} + \log \mathcal{T}(y_{next} | y) + \log \mathcal{P}(x_j | y)] \\
 S_{i,j,y}(y = DEL) &= \max_{y_{next}} [S_{i,j,y_{next}} + \log \mathcal{T}(y_{next} | y)] \\
 S_{i,j,y}(y = BIFURC) &= \max_{i-1 <= mid <= j} [S_{i,mid,y_{left}} + S_{mid+1,j,y_{right}}]
 \end{aligned}$$

At the end, the score of the global alignment is in $S_{1,N,1}$. The alignment itself is reconstructed by tracing back through the matrix beginning at $S_{1,N,1}$, as is usual for dynamic programming methods, following the maximum-scoring path at each state. The algorithm requires $O(N^2M)$ memory and $O(N^3M)$ time. It takes roughly four megabytes of memory and one or two seconds to align a tRNA sequence to a tRNA CM on a Silicon Graphics R4000 Indigo.

The score is reported as a log-odds score, by subtracting from the log likelihood of the alignment the log likelihood that the sequence was generated as random sequence of equiprobable base composition. Scores are reported in log base two, i.e. in bits. This can be done simply by precalculating log-odds scores in place of log likelihood scores for each symbol emission probability, prior to alignment. Log odds scoring corrects for the strong length dependence of log likelihood scores. This correction makes database searching possible and greatly simplifies interpretation of scores. Scores above zero are more likely matches to the model than to random sequence, and the more positive the better.

The alignment algorithm is similar to the Nussinov/Zuker algorithms for folding individual RNAs [38, 39] and to the Needleman/Wunsch algorithm for aligning two primary sequences. It is also very similar to the Inside-Outside algorithm proposed for alignment of stochastic context-free grammars to speech data [40, 41], except that we make the Viterbi assumption that the probability of the model emitting the sequence is approximately equal to the probability of the single best alignment of model to sequence, rather than the sum of the probabilities of all possible alignments [36]. The Viterbi assumption conveniently produces a single optimal alignment rather than a probability distribution over all possible alignments.

Model training

Given a set of example sequences, we want to find the CM which has the maximum likelihood of generating those sequences. This is model “training” (Figure 3). It is a global optimization problem with no practical rigorous solution. In fact we have two problems: deciding on a structure for the model (how many nodes and states and how they branch), and finding optimal values for the state transition probabilities and symbol emission probabilities given that structure. The second problem is soluble by methods such as expectation maximization (EM), which find good local optima for parameter values. That leaves us the problem of consensus secondary structure determination.

Heuristic methods have been proposed for RNA consensus secondary structure prediction based on correlations observed in multiple alignments [42, 43]. Instead, we present an optimal and efficient dynamic programming algorithm for consensus RNA secondary structure prediction from RNA sequence alignments, and we use this algorithm for model construction. The algorithm uses values of mutual information content for all pairs of columns in a multiple alignment [42, 31]. In a multiple sequence alignment, the mutual information of column i and column j , where x_i range over possible symbols A, C, G, and U, f_{x_i} is the independent frequency of a symbol in column i , and $f_{x_i x_j}$ is the joint frequency of a symbol pair in columns i and j , is:

$$M_{i,j} = \sum_{x_i, x_j} f_{x_i x_j} \log_2 \frac{f_{x_i x_j}}{f_{x_i} f_{x_j}}$$

$M_{i,j}$ varies from 0 bits (no correlation) to 2 bits of mutual information (perfectly correlated base pair with no primary sequence conservation). The $M_{i,j}$ values of the master alignment of 1415 tRNA sequences are shown in Figure 4. The mutual information represents an expected gain in score from assigning columns i and j to a pairwise state instead

of to singlet states. In general, the consensus secondary structure will be included in the tree which captures as much correlation information as possible. This tree is calculated by applying a dynamic programming calculation to the pairwise mutual information scores, starting from the diagonal $i = j$ and working towards $i = 1, j = N$ and using the recursion:

$$S_{i,j} = \max[S_{i+1,j}, S_{i,j-1}, S_{i+1,j-1} + M_{i,j}, \max_{i < mid < j}[S_{i,mid} + S_{mid+1,j}]]$$

This is the Nussinov/Zuker dynamic programming RNA folding algorithm [38, 44] except that the score being optimized is a function of mutual information terms rather than of number of base-pairs or of thermodynamic stacking energies. The time and memory required are negligible relative to the alignment algorithm. $S_{1,N}$ will contain the maximal sum of the covariations $M_{i,j}$ that can be captured by a tree, and the traceback of the score matrix produces the structure of that optimal tree (Figure 4). An approximate covariance model structure is derived from this tree. Columns of the alignment are assigned to match nodes or insert states of neighboring nodes according to how many symbols occupy the column, using an arbitrary threshold of 50%. By definition, the new model is aligned to all of the training sequences in the multiple alignment, so symbol emission and state transition parameters \mathcal{P} and \mathcal{T} are calculated using the re-estimation procedure described below.

Given an initial model, we find locally optimal values for the state transition probabilities \mathcal{T} and symbol emission probabilities \mathcal{P} by iterative re-estimation, using the Viterbi approximation to Baum-Welch expectation maximization (EM) [10, 36]. Each example sequence is aligned to the current model using the alignment algorithm. Re-estimates for parameters \mathcal{P} and \mathcal{T} are made from these alignments based on the observed counts of state transitions and symbol emissions, $n(y_{next} | y)$ and $n(x | y)$, using a procedure like that described in [10]:

$$\begin{aligned} \mathcal{P}(x | y) &= \frac{n(x | y) + \mathcal{R}(x | y)}{n(y) + \sum_{x'=A,C,G,U} \mathcal{R}(x' | y)} \\ \mathcal{T}(y_{next} | y) &= \frac{n(y_{next} | y) + \mathcal{R}(y_{next} | y)}{n(y) + \sum_{y_{next}} \mathcal{R}(y_{next} | y)} \end{aligned}$$

This corresponds to a Bayesian mean posterior probability estimate given a Dirichlet prior with parameters \mathcal{R} . If the parameters \mathcal{R} are equal to one, the equations correspond just to a standard small sample statistics correction of the observed frequencies, Laplace's law of succession [45]; we use this for match state emissions. Following [10], we use high \mathcal{R} values to fix insert state emissions to be equiprobable regardless of observed counts, and we

also bias state transition parameters \mathcal{R} to favor transitions into MATP > MATL, MATR > INSL, INSR, DEL. This is a “subjective” or expert Bayesian prior. Alternatively, we could derive prior probability parameters \mathcal{R} from RNA alignment data. The alignment and re-estimation process is repeated until values for the parameters \mathcal{P} and \mathcal{T} converge. The process is guaranteed to converge to a local optimum [10, 36].

Therefore, a full training procedure is as follows (Figure 3). An initial model is created from a (possibly random) alignment of the example sequences using the model construction algorithm. The model’s symbol emission and state transition probabilities \mathcal{P} and \mathcal{T} are iteratively re-estimated by an EM algorithm. When the parameters converge, a new model is built from the current multiple sequence alignment with the model construction algorithm. This cycle is repeated until neither the model structure nor its parameters are changing significantly.

This training procedure is analogous to the intuitive manual process of comparative sequence analysis. A starting alignment is refined iteratively as progressively more correlations and conserved positions are perceived. Training obviously works best if the starting alignment is good. Importantly, though, we have found that it also works when started from random alignments of example sequences.

Searching

The searching algorithm for finding high-scoring subsequences in an arbitrarily long sequence is nearly identical to the alignment algorithm. The search scoring matrix is indexed by distance from diagonal d , j , and y instead of i , j , y . Scanning across a long sequence is achieved by adding a new row j for the next sequence position, calculating scores starting from the diagonal $d = 0$, and examining the final scores at $y = 1$ for each d . These scores are the scores of alignments to all the possible subsequences that end in x_j . The starting point of a match is known ($i = j - d$) without having to do a traceback of the scoring matrix. By restricting the subsequences scored to a certain maximum length w and using a matrix index $j' = j \bmod w$ instead of j , the scoring calculation maintains constant memory size regardless of the length of the sequence being searched.

tRNA data sets

The 1993 compilation of aligned tRNA sequences [12] was obtained from the ftp.embl-heidelberg.de anonymous ftp server. The 87 noncanonical “group III” sequences were

removed, as well as the 509 RNA sequences (which are often redundant with the DNA sequences in the database), leaving a database of 1415 aligned tRNA DNA sequences. Of these, 62 are archaeobacterial, 242 are eukaryotic nuclear, 259 are from chloroplasts, 249 are eubacterial, 579 are mitochondrial, and 24 are viral. This alignment was assumed to be correct, and is referred to throughout as the “trusted” alignment.

100 sequences were selected at random from this database to form an independent TEST100 test sequence set. A training set of 100 sequences, SIM100, was selected randomly from the 1315 remaining sequences. An additional training set of 100 sequences, SIM65, was constructed by filtering out similar sequences; the 1315 sequences were clustered by pairwise aligned primary sequence identity and then all but one homologous sequence was removed at random from clusters over a 65% average similarity threshold.

Implementation

The algorithms were implemented in ANSI C on a Silicon Graphics R4000 Indigo. The programs are known to be portable across several UNIX architectures, including Silicon Graphics, Sun, DEC Alpha, MIPS M2000, and Alliant FX/2800 machines. The software, parameter sets, and tRNA alignments are available via anonymous ftp from cele.mrc-lmb.cam.ac.uk (131.111.84.1) or by request from S.R.E.

Results

Training and test tRNA data sets

Transfer RNA (tRNA) is ideal for testing these algorithms. Well over a thousand tRNA sequences are known. Although their primary sequences vary, almost all tRNAs share a common structure. Multiple tRNA sequence alignments are the only available RNA alignments that utilize X-ray crystal structure information. A compilation of aligned tRNA sequences is freely available [12]. We obtained a master trusted alignment of 1415 tRNA sequences from this database (see Methods). 100 sequences were held out as independent test data.

We picked two training sets of 100 sequences each. One set, SIM100, was selected randomly from the 1315 training sequences. The second set, SIM65, was created as described in Methods and contains 100 particularly dissimilar tRNA sequences. The most related pair of sequences in SIM65 is 69% identical when correctly aligned. The average pairwise

identity in all the data sets is between 35% and 40% (Table 1). The accuracy of standard pairwise sequence alignment [46] begins to sharply drop off for pairs of tRNA sequences less than about 65% identical (data not shown). ClustalV, a popular and reliable multiple sequence alignment program [47], produces poor alignments for all these data sets which range from 37% to 63% identical to the trusted alignments. This is almost as bad as one gets from uninformative alignments; removing all gaps from the sequences gives “alignments” which are about 30% identical to the correct alignment (Table 1). [We measure alignment identity as the fraction of aligned symbol pairs in the trusted alignment that are also aligned in the other alignment.]

Given a multiple sequence alignment, an information content [48] can be calculated for the primary sequence consensus, and estimates of the pairwise second-order information content can be obtained. These information measures indicate how much extra information may be gained by models which capture pairwise second-order information. There is almost as much information in the secondary structure of tRNA as in the primary sequence consensus (Table 1). A secondary structure representation of tRNA such as a CM should use twice as much information about tRNA sequences as a primary sequence representation such as an HMM or a profile.

Table 1 also shows an estimate of how much additional pairwise information is available from tertiary contacts that a CM does not capture. We calculated an upper bound on the total pairwise correlation information that includes all pairwise contacts, not just those consistent with classical nested secondary structure. This number is less than 10% greater than the figure for secondary structure (Table 1). A CM captures over 90% of all pairwise information in tRNA sequences.

Consensus structure prediction

Because a secondary structure is implicit in the structure of a covariance model, covariance model training can be used to predict a consensus RNA secondary structure. The model training algorithm derives a consensus secondary structure prediction directly from an alignment, or from initially unaligned sequences (Figure 3).

The easier problem is to build a model from an existing alignment. Since trusted structural alignments already exist for many important biological RNAs, this ability to go straight from an alignment to a model for searching databases is useful. CMs were trained starting from the trusted alignments of each of the training sets. These models (the A models) converged rapidly and had average scores of from 46.7 bits to 58.7 bits

(Table 2). Hidden Markov models, which use primary sequence information alone, reach average scores of from 22 to 30 bits trained on the same alignments (data not shown). As expected, a covariance model captures about twice as much information as an HMM. The scores are significant fractions of the predicted upper bounds (60-70 bits; see Table 1); the difference is due to the costs associated with the model's state transition probabilities (entropy due to permissible variations in structure).

The harder problem is to create a model from unaligned and unfolded sequences, where the correct consensus structure and alignment are initially unknown. Models were trained starting from unaligned training sets and no information about the structure of tRNA. After 13 to 29 total rounds of iteration and 2 to 5 model structure changes, these models (the U models) converged at final scores of 47.2 to 56.7 bits (Table 2). These values are comparable to the results for models started from the trusted alignments.

The alignment of the U models to the sequence of yeast tRNA-Phe was examined to see if the models were correctly assigning the cloverleaf secondary structure. The pairwise assignments of the model should include the pairwise interactions in the secondary and tertiary structure. Indeed, the resulting secondary structure prediction of tRNA-Phe by both models was entirely correct. Two tertiary contacts are consistent with the non-pseudoknotting constraint (G_{26} - A_{44} and U_{54} - A_{58} [49]) and these were also predicted by both U models.

Multiple sequence alignment

The A models and the U models were used to produce multiple sequence alignments of the 100 independent test sequences, and these alignments were compared to the trusted alignment [12]. The A models produced alignments ranging from 93% to 95% correct. The U models produced alignments ranging from 90% to 92% correct (Table 2). For example, trusted and predicted alignments for a small set of five tRNA sequences are shown in Figure 5.

A pseudo-random alignment produced by just removing all the gaps from the test set is 30% correct. A rough upper bound was also estimated; since only a few tRNA crystal structures are known, the "correct" alignment is not unambiguous and it would be fairly suspicious if a method did achieve 100% identity to the trusted alignment. A careful independent human alignment (S.R.E.) of a randomly chosen set of ten of the test sequences scored 97% identity to their trusted alignment. Despite their ignorance of most tertiary structure information, CMs can use secondary structure information to produce

alignments very nearly as good as human alignments.

Database searching

One major goal of CMs is to provide a general-purpose RNA searching tool, obviating the need for custom-built programs for every different RNA family. We therefore tested how well a CM performs relative to one of the carefully crafted tRNA detection programs that are available [22, 24, 25]. The best is probably Fichant and Burks' TRNASCAN [22]. In a search of GenBank release 66, TRNASCAN detected 725 of 744 known non-organelle tRNAs (97.5% true positive rate), 26 false positives in 69.2 Mb searched (0.37 false positives/Mb), and 16 previously unnoticed probable tRNAs [22].

For searching, we used the best tRNA model (A1415), trained from the trusted alignment of all 1415 tRNA sequences. A1415 was compared to the GenBank structural RNA database (Release 72, 1.9 Mb) using the database scanning version of the alignment algorithm and both strands of 2.2 Mb of *C. elegans* genomic sequence [4]. 15 possible tRNAs are suggested by TRNASCAN in the *C. elegans* genomic sequence. By human inspection, one of these predictions appears to be a false positive; 14 of the 15 have been annotated as tRNA genes (Erik Sonnhammer, personal communication).

These data are summarized in Figure 6. Fichant and Burks trained and tested on only a subset of more well-conserved "cytoplasmic" tRNAs, from eukaryotic cytoplasm, archaeobacteria, or eubacteria, and excluded less conserved "other" tRNAs such as mitochondrial or selenocysteine tRNAs. It was difficult for us to perform this separation cleanly using the annotations of the 522 tRNA sequences in the GenBank structural RNA database. Instead, we counted the scores of the 1415 sequences of the well-annotated tRNA database as the true positives and split them into 547 "cytoplasmic" tRNAs and 868 "other" tRNAs to facilitate comparison to TRNASCAN. These are scores to the training set itself; only the *C. elegans* genomic tRNAs constitute an independent test set in this experiment. The non-tRNAs, providing an estimate of the false positive rate, are from non-tRNA scores from the GenBank structural RNA database and from the *C. elegans* genomic sequence (6.3 Mb total).

Any score cutoff between 11.7 and 25.9 cleanly separates all the non-tRNAs from the 547 cytoplasmic RNAs, giving a sensitivity of > 99.98% and a false positive rate of <0.2/Mb, compared to TRNASCAN's 97.5% sensitivity and 0.37/Mb false positive rate. There are two "non-tRNA" hits in the GenBank database at scores of 14.4 and 32.9 which are due to repetitive elements known as R.dre.1 or identifier (ID). These are members of a family of

SINEs (short interspersed nuclear elements) in rodents which are thought to have originated from tRNA [50]. They are not detected by TRNASCAN. We did not count these as false positives.

As a further test of searching a genome for relatively non-canonical tRNA genes, we also ran both TRNASCAN and the covariance model A1415 on the complete mitochondrial genome of *Podospira anserina* (PANMTPACGA), which is annotated as having 27 tRNA genes. A1415 detects 27/27 (100%) of them with no false positives for a cutoff between 15 and 23 bits; the highest non-tRNA hit, at 15 bits, is highly AT-rich and within a coding sequence, and is thus probably a real negative. TRNASCAN detects 18/27 (67%) of them with no false positives. 21 of the 27 *Podospira* tRNA sequences were not in the A1415 training set.

There are tRNAs that are difficult to recognize. 33/868 (5%) of the “other” tRNAs score below 12 bits. 31 of these are mitochondrial; the other two are phage T5 tRNA-Ser and a *D. melanogaster* selenocysteine tRNA. In the GenBank structural RNA database, 26/522 (5%) of annotated tRNAs were missed. 22 of the 26 missed GenBank tRNAs are *Ascaris suum* mitochondrial tRNAs which completely lack the dihydrouridine stem-loop [51]. The remaining four were mitochondrial tRNA-Ser from human, hamster, and cow, and a yeast suppressor tRNA (YSCLSC).

In *C. elegans* genomic sequence, all 14 putative tRNA genes were detected. 12 intronless tRNA genes give scores of 63.5 to 77.0, and two intron-containing tRNA genes give scores of 31.6 and 31.7. The 15th tRNA proposed by TRNASCAN and subsequently rejected after human inspection scored -42.5 and was also rejected by the A1415 CM. The detection of the intron-containing tRNA genes demonstrates the flexibility of the model, which had only seen intronless sequences during training.

Discussion

We describe a “covariance model”, a general probabilistic model of RNA secondary structure and sequence consensus. A CM allows insertions, deletions, and mismatches relative to the consensus, assigning them scores based on probabilities observed in example RNAs. Base pairs are scored in a manner that allows any combination of primary sequence conservation and pairwise correlation with another sequence position. Any type of pairwise correlation, canonical Watson-Crick or other noncanonical interactions, can be scored. Previously, it has only been possible to make probabilistic models of primary sequence con-

sensus [7, 8, 10]. RNA structures have been modeled either with custom-built programs [22, 23, 24, 25] or with fairly inflexible non-probabilistic descriptions [27, 28]. Full probabilistic descriptions of a molecule have very significant advantages in database searching. Probabilistic models incorporate information from even weakly conserved features and have superior sensitivity and discrimination compared to more deterministic pattern-searching methods. We find that a tRNA CM is superior even to a custom-built tRNA searching program that incorporates partial probabilistic information [22]. Also, a probabilistic model can be flexible enough to recognize unexpectedly related sequences. Good examples of this were given when a tRNA model recognized two R.dre.1 SINE sequences in GenBank’s structural RNA database which are members of a repetitive element family thought to be derived from tRNA [50], and when a tRNA model recognized intron-containing tRNAs in *C. elegans* genome sequence although it had been trained only on intronless tRNAs. CMs give us the ability to search for homologues of RNAs that conserve only small amounts of primary sequence. We are interested in constructing models of other RNA families, especially of the various catalytic RNAs [52, 23, 53, 54] in order to search for unnoticed examples of these molecules in the sequence databases.

We build CMs directly from RNA sequence alignments, when such alignments are available. No additional structural information is necessary to produce accurate secondary structure predictions when the alignment is given, because strong covariances make the correct structure obvious. The model construction procedure uses a fast and efficient dynamic programming algorithm to find a globally optimal model structure. This structure consists of the consensus secondary structure plus a few additional tertiary interactions that happen to be compatible with the non-pseudoknotting restriction. Non-optimal, heuristic methods have been proposed before for RNA structure prediction from alignments [42, 43]. We believe ours is the first globally optimal consensus secondary structure prediction algorithm that has been proposed. We are currently using the model construction algorithm on its own to rapidly analyze alignments of families of DNA repeat sequences in the *C. elegans* genome to see if any secondary structure is apparent, since some classes of mammalian SINE elements are apparently derived from structural RNA transcripts.

We also find that models can be trained from initially unaligned sequences, using no prior information about the consensus structure of the family, which means that CMs can be used for consensus secondary structure prediction. This represents a fundamentally new computational technique for RNA consensus structure prediction. Previous efforts have largely concentrated on calculating thermodynamically optimal and suboptimal individual structures using the Zuker/Nussinov RNA folding algorithms [38, 39], then searching for

common structures amongst the alternative foldings for each sequence [55, 34]. Instead, CMs are essentially an automatic implementation of the comparative sequence analysis methods that have been instrumental in producing the accepted consensus secondary structures of numerous RNA families [56, 57, 31, 16, 53, 58, 59, 60, 61]. CMs could be used to seek a consensus structure of RNA sequence families for which such information is not yet known, such as some of the nucleolar snRNA families [62].

We also find that CM training can produce multiple sequence alignments of quite high accuracy. In general, RNA alignments have been produced by hand because their accuracy relies so heavily on pairwise correlations. The only described simultaneous folding and multiple alignment algorithm, that of Sankoff [63], which finds a globally optimum multiple alignment that optimizes a linear combination of thermodynamic folding energy and primary sequence alignment score, has not been practical to implement. We briefly considered the possibility that CM-generated alignments are better than the trusted human ones. We checked the alignments of yeast phenylalanine and yeast aspartate tRNAs against a structural alignment produced by superposition of the two crystal structures (data not shown), and verified that the trusted alignment was correct and that the CM alignment tended to be inaccurate in regions where correct alignment required tertiary structural information (see Figure 5).

Because CMs ignore additional covariation-inducing tertiary structural interactions such as pseudoknots, our methods can only be an aid, not a complete solution, to producing the highly accurate multiple sequence alignments necessary for low resolution three-dimensional RNA structure prediction problems [15]. However, the contribution of tertiary interactions is not crucial for database searching purposes. We show (Table 1) that tertiary structure contributes at most two or three bits of pairwise correlation information to tRNAs, compared to 30-40 bits in primary sequence consensus and 30 bits of secondary structure pairwise correlation information. We expect these rough proportions to be about the same for most RNAs; pseudoknots may be functionally important in RNA structures but they usually account for relatively few base pairs. We are exploring methods to deal with pseudoknots but these will have to be either heuristic additions on top of our algorithms or quite different from the CM framework we describe, because the exclusion of pseudoknotted interactions is enforced both by the structure of a CM and by the dynamic programming algorithms we use.

Our approach was conceived as a synthesis of the application of hidden Markov modeling to protein sequences by Krogh *et al.* [10], and the Zuker/Nussinov algorithms for finding thermodynamically optimal foldings of individual RNAs [38, 39]. Later, we discovered that

the step from HMMs to CMs was well known in formal language theory [29]. HMMs are stochastic regular grammars, and our CMs could be described as stochastic context-free grammars (SCFGs), one step more general in the Chomsky hierarchy of formal grammars [64, 65]. Our alignment and training algorithms are closely akin to the algorithms for using SCFGs to model speech [40, 41]. Searls has already proposed non-stochastic context-free grammars as an RNA modeling tool [29]. While our work was in progress, we also became aware of work by Sakakibara *et al.* [30] which introduces similar RNA methods that are more closely faithful to the stochastic context-free grammar formalism. They describe an elegant and fast training method that takes advantage of base-pairing information when it is already known, and they build RNA SCFGs manually from prior knowledge of an RNA secondary structure. In contrast, our model training algorithms build CM structures fully automatically and let us work quickly and easily from either an existing sequence alignment or even from unaligned and unfolded sequences.

The most serious drawback of these methods is that the size of RNAs that we can deal with is sharply limited by the computational demands of the alignment algorithm. We currently cannot analyze sequences much longer than 150-200 nucleotides. Database search speed is about 10-20 bases/second for tRNA models, which is painfully slow for full-scale searching of entire sequence databases. For now, the programs are sufficient to build models of small snRNA, tRNA, and repeat families and keep pace with the analysis of the *C. elegans* genome sequencing project. We are exploring workarounds for studying larger RNA molecules. Another drawback is that a large number of sequences are required to build a good model. Although consensus structure prediction and multiple alignment can be reasonably accurate with as few as ten or twenty sequences (data not shown), a satisfyingly discriminative model for searching databases can require a hundred or more sequences. We hope to alleviate this problem somewhat by more sophisticated incorporation of prior knowledge about RNA structure into CM parameter estimation, guided by Bayesian methods [10, 66]. Our current Bayesian prior probability distributions are subjective rather than derived directly from known RNA alignments.

In vitro RNA evolution and selection techniques have been devised to select novel small RNA sequences for particular functions, often for protein binding but also for catalysis. These techniques are being used to study the catalytic and structural repertoire available to RNA [21, 19, 33, 20], and for the “irrational” [67] design of potential pharmaceuticals. The consensus structure and sequence of the resulting small RNA families is often apparent to the eye, but can sometimes be rather elusive. We expect that CMs may prove especially suited to the analysis of sequence families generated by such RNA selection and evolution

experiments.

Acknowledgements

S.R.E. was supported by a long-term postdoctoral fellowship from the Human Frontier Science Program. We thank Graeme Mitchison, Donna Albertson, Jim Haseloff, and David MacKay for critical reading of the manuscript, Erik Sonnhammer for helpful discussions, and Jim Reed and NeXagen, Inc., Boulder, Colorado, for some of our computer time.

References

- [1] Bork, P., Ouzounis, C., Sander, C., Scharf, M., Schneider, R., and Sonnhammer, E. (1992) *Protein Science* **1**, 1677–1690.
- [2] Green, P., Lipman, D., Hillier, L., Waterston, R., States, D., and Claverie, J.-M. (1993) *Science* **259**, 1711–1716.
- [3] Oliver, S., van derAart, Q., Agostoni-Carbone, M., Aigle, M., et al. (1992) *Nature* **357**, 38–46.
- [4] Sulston, J., Du, Z., Thomas, K., Wilson, R., Hillier, L., Staden, R., Halloran, N., Green, P., Thierry-Mieg, J., Qiu, L., Dear, S., Coulson, A., Craxton, M., Durbin, R., Berks, M., Metzstein, M., Hawkins, T., Ainscough, R., and Waterston, R. (1992) *Nature* **356**, 37–41.
- [5] Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990) *J. Mol. Biol.* **215**, 403–410.
- [6] Pearson, W. and Lipman, D. (1988) *Proc. Natl. Acad. Sci. USA* **85**, 2444–2448.
- [7] Barton, G. J. (1990) *Meth. Enzymol.* **183**, 403–427.
- [8] Gribskov, M., Luthy, R., and Eisenberg, D. (1990) *Meth. Enzymol.* **183**, 146–159.
- [9] Baldi, P., Chauvin, Y., Hunkapiller, T., and McClure, M. A. (1994) *Proc. Natl. Acad. Sci. USA* **91**, 1059–1063.
- [10] Krogh, A., Brown, M., Mian, I. S., Sjolander, K., and Haussler, D. (1994) *J. Mol. Biol.* **235**, 1501–1531.
- [11] Paul R. Schimmel, Dieter Soll, and John N. Abelson, (ed.) (1979) *Transfer RNA: Structure, Properties, and Recognition*, Cold Spring Harbor Laboratory Press, New York.
- [12] Steinberg, S., Misch, A., and Sprinzl, M. (1993) *Nucl. Acids Res.* **21**, 3011–3015.
- [13] Larsen, N. and Zwieb, C. (1993) *Nucl. Acids Res.* **21**, 3019–3020.
- [14] Lambowitz, A. M. and Belfort, M. (1993) *Ann. Rev. Biochem.* **62**, 587–622.

- [15] Michel, F., Netter, P., Xu, M.-Q., and Shub, D. (1990) *Genes Dev.* **4**, 777–788.
- [16] Guthrie, C. and Patterson, B. (1988) *Ann. Rev. Genet.* **22**, 387–419.
- [17] Rosen, C. A. (1991) *Trends Genet.* **7**, 9–14.
- [18] Theil, E. C. (1990) *J. Biol. Chem.* **265**, 4771–4774.
- [19] Ellington, A. and Szostak, J. (1990) *Nature* **346**, 818–822.
- [20] Tuerk, C. and Gold, L. (1990) *Science* **249**, 505–510.
- [21] Bartel, D. P. and Szostak, J. W. (1993) *Science* **261**, 1411–1418.
- [22] Fichant, G. A. and Burks, C. (1991) *J. Mol. Biol.* **220**, 659–671.
- [23] Lisacek, F., Diaz, Y., and Michel, F. (1994) *J. Mol. Biol.* **235**, 1206–1217.
- [24] Marvel, C. C. (1986) *Nucl. Acids Res.* **14**, 431–435.
- [25] Staden, R. (1980) *Nucl. Acids Res.* **8**, 817–825.
- [26] Bairoch, A. (1993) *Nucl. Acids Res.* **21**, 3097–3103.
- [27] Gautheret, D., Major, F., and Cedergren, R. (1990) *Comput. Applic. Biosci.* **6**, 325–331.
- [28] Saurin, W. and Marliere, P. (1987) *Comput. Applic. Biosci.* **3**, 115–120.
- [29] Searls, D. B. (1992) *American Scientist* **80**, 579–591.
- [30] Sakakibara, Y., Brown, M., Underwood, R. C., Mian, I. S., and Haussler, D. (1994) In Hunter, L. (ed.), *Proceedings of the Twenty-Seventh Annual Hawaii International Conference on System Sciences: Biotechnology Computing*, volume **V**, Los Alamitos, CA: IEEE Computer Society Press. pp. 284–293.
- [31] Gutell, R., Power, A., Hertz, G., Putz, E., and Stormo, G. (1992) *Nucl. Acids Res.* **20**, 5785–5795.
- [32] Woese, C. R. and Pace, N. R. (1993) In Gesteland, R.F. and Atkins, J.F. (ed.), *The RNA World*, Cold Spring Harbor Laboratory Press New York.
- [33] Robertson, D. and Joyce, G. (1990) *Nature* **344**, 467–468.

- [34] Shapiro, B. A. and Zhang, K. (1990) *Comput. Applic. Biosci.* **6**, 309–318.
- [35] tenDam, E., Pleij, K., and Draper, D. (1992) *Biochemistry* **31**, 11665–11676.
- [36] Rabiner, L. R. (1989) *Proc. IEEE* **77**, 257–286.
- [37] Knuth, D. E. (1973) *The Art of Computer Programming: Fundamental Algorithms*, volume 1, Addison–Wesley, Reading, Massachusetts second edition.
- [38] Nussinov, R., Pieczenik, G., Griggs, J. R., and Kleitman, D. J. (1978) *SIAM J. Appl. Math.* **35**, 68–82.
- [39] Zuker, M. and Stiegler, P. (1981) *Nucl. Acids Res.* **9**, 133–148.
- [40] Lari, K. and Young, S. (1990) *Computer Speech and Language* **4**, 35–56.
- [41] Lari, K. and Young, S. (1991) *Computer Speech and Language* **5**, 237–257.
- [42] Chiu, D. K. and Kolodziejczak, T. (1991) *Comput. Applic. Biosci.* **7**, 347–352.
- [43] Han, K. and Kim, H.-J. (1993) *Nucl. Acids Res.* **21**, 1251–1257.
- [44] Zuker, M. (1989) *Science* **244**, 48–52.
- [45] Berg, O. G. and vonHippel, P. H. (1987) *J. Mol. Biol.* **193**, 723–750.
- [46] Needleman, S. and Wunsch, C. (1970) *J. Mol. Biol.* **48**, 443–453.
- [47] Higgins, D., Bleasby, A., and Fuchs, R. (1992) *Comput. Applic. Biosci.* **8**, 189–191.
- [48] Schneider, T. D., Stormo, G. D., Gold, L., and Ehrenfeucht, A. (1986) *J. Mol. Biol.* **188**, 415–431.
- [49] Kim, S.-H. (1979) In Schimmel, P.R., Soll, D., and Abelson, J.N. (ed.), *Transfer RNA: Structure, Properties, and Recognition*, Cold Spring Harbor Laboratory New York.
- [50] Daniels, G. R. and Deininger, P. L. (1985) *Nature* **317**, 819–822.
- [51] Okimoto, R. and Wolstenholme, D. R. (1990) *EMBO J.* **9**, 3405–3411.
- [52] Cech, T. R. and Bass, B. L. (1986) *Ann. Rev. Biochem.* **55**, 599–629.
- [53] Michel, F., Umesono, K., and Ozeki, H. (1989) *Gene* **82**, 5–30.

- [54] Symons, R. H. (1992) *Ann. Rev. Biochem.* **61**, 641–671.
- [55] Konings, D. and Hogeweg, P. (1989) *J. Mol. Biol.* **207**, 597–614.
- [56] Brown, J. W., Haas, E. S., James, B. D., Hunt, D. A., and Pace, N. R. (1991) *J. Bacteriol.* **173**, 3855–3863.
- [57] Fox, G. E. and Woese, C. R. (1975) *Nature* **256**, 505–507.
- [58] Michel, F. and Westhof, E. (1990) *J. Mol. Biol.* **216**, 585–610.
- [59] Noller, H. F. and Woese, C. R. (1981) *Science* **212**, 403–411.
- [60] Noller, H. F., Kop, J., Wheaton, V., Brosius, J., Gutell, R. R., Kopylov, A. M., Dohme, F., Herr, W., Stahl, D. A., Gupta, R., and Woese, C. R. (1981) *Nucl. Acids Res.* **9**, 6167–6189.
- [61] Zwieb, C. (1989) *Prog. Nucl. Acid Res. Mol. Biol.* **37**, 207–234.
- [62] Fournier, M. J. and Maxwell, E. S. (1993) *Trends Biochem. Sci.* **18**, 131–135.
- [63] Sankoff, D. (1985) *SIAM J. Appl. Math.* **45**, 810–825.
- [64] Chomsky, N. (1959) *Information and Control* **2**, 137–167.
- [65] Gersting, J. L. *Mathematical Structures for Computer Science* chapter 8 W.H. Freeman New York, NY (1993).
- [66] MacKay, D. J. (1992) *Neural Computation* **4**, 415–447.
- [67] Brenner, S. and Lerner, R. (1992) *Proc. Natl. Acad. Sci. USA* **89**, 5381–5383.

Figure legends

Figure 1

A. An example RNA structure. B. An ordered binary tree description of that RNA structure. The tree includes dummy begin, end, and branching (bifurcation) nodes in addition to pairwise and singlet nodes that account for sequence.

Figure 2

The seven distinct types of nodes from Figure 1 are broken up into states as shown. There are seven different kinds of states in all (bifurcation BIF, begin BEG, insert-left INSL, insert-right INSR, match-pairwise MATP, match-left MATL, match-right MATR, and delete DEL). State transition probabilities are indicated by arrows. States which have singlet or pairwise symbol emission probabilities are indicated by “ACGU” beside the state.

Figure 3

The covariance model training algorithm.

Figure 4

The half-diagonal matrix of $M_{i,j}$ mutual information values for the trusted alignment of all 1415 tRNA sequences. X and Y coordinates are numbered according to the canonical scheme for tRNA positions. Values of 0.0 to 1.0 bits are squares colored in linear grey scale and values of greater than 1.0 bits are black squares. The path of the traceback from the model construction algorithm is superposed on the matrix. Thick lines run through assigned nodes and thin lines connect the bifurcation points. The arrow indicates an example of a strong covariance from the tertiary contact G₁₅-C₄₈ in the yeast tRNA-Phe structure which the non-pseudoknotting restriction prevents the model from including. The structure, canonical numbering scheme, and tertiary contacts (dashed lines) of yeast tRNA-Phe are also shown.

Figure 5

Multiple sequence alignment of five tRNA sequences whose three-dimensional structures are known from X-ray crystallography. Top, the trusted structural alignment [12]; middle, the alignment produced by model U100; bottom, the alignment produced by a primary sequence alignment algorithm, ClustalV [47]. The nucleotides in lower case in the U100 alignment are those assigned to insert states. The nucleotides in the canonical cloverleaf secondary structure are in grey. DF6280, yeast tRNA-Phe (PDB1TRA); “DF6280G”, genomic sequence of a yeast tRNA-Phe (GenBank YSCTGFT15); DD6280, yeast tRNA-Asp (PDB2TRA); DX1661, *E. coli* initiator tRNA-fMet (PDB0FMT); DS6280, yeast tRNA-Ser

(PDB5TRA). The genomic sequence of tRNA-Phe is included as an example of the ability of the U100 CM to accommodate deviations from the structures in its training set.

Figure 6

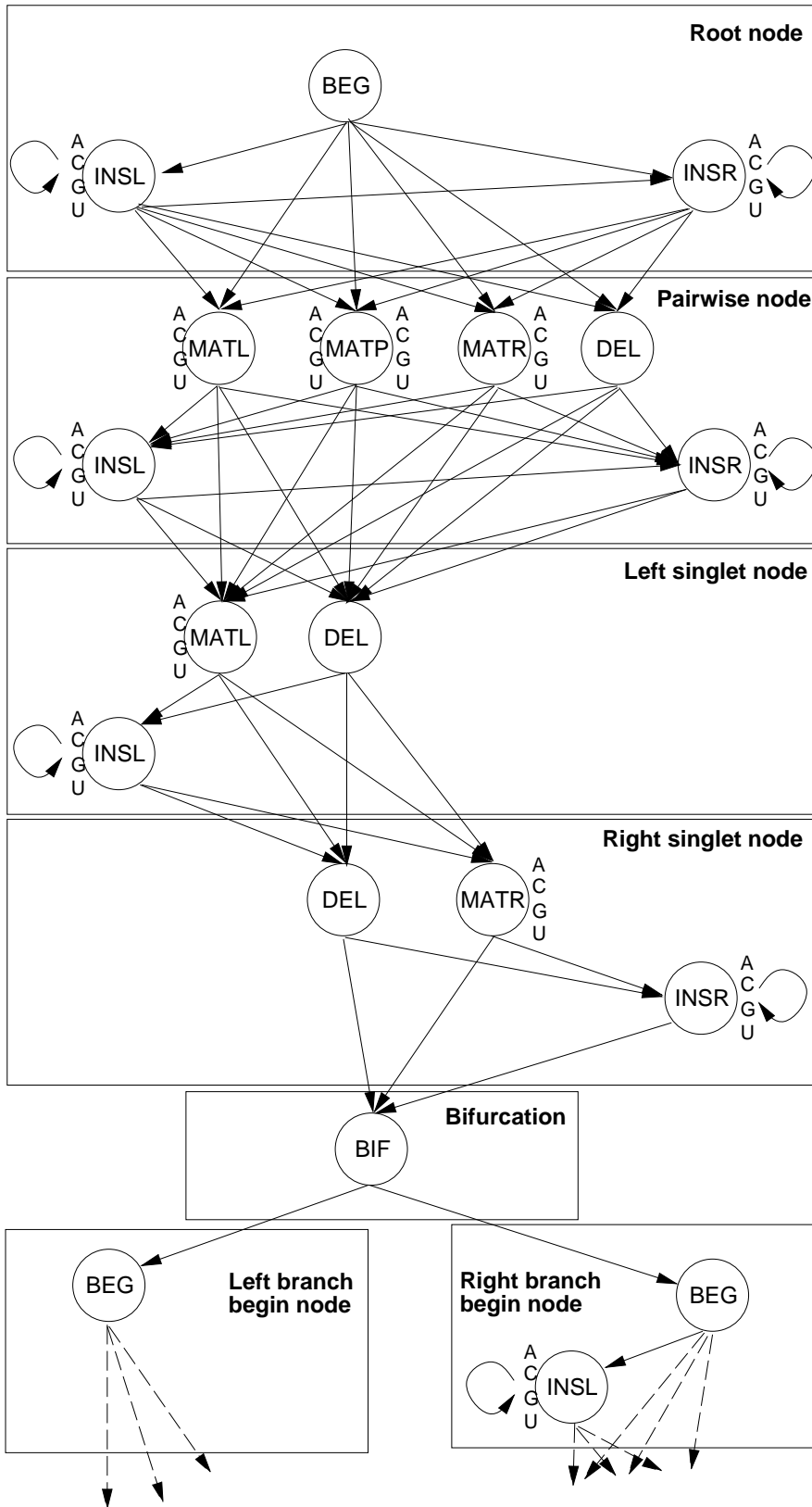
Number of hits versus score in bits, from using the A1415 model to search a total of 6.3 Mb of sequence from the Genbank structural tRNA database and both strands of the current genomic *C. elegans* sequence. In white are the background of non-tRNA hits. Hits to 547 non-selenocysteine “cytoplasmic” tRNAs are in black. “Other” tRNA hits, in grey, are the scores of 868 mitochondrial, chloroplast, viral, and selenocysteine tRNAs. Arrows indicate the gap between the highest non-tRNA hit (with two tRNA-related exceptions; see text) and the lowest scoring cytoplasmic tRNA.

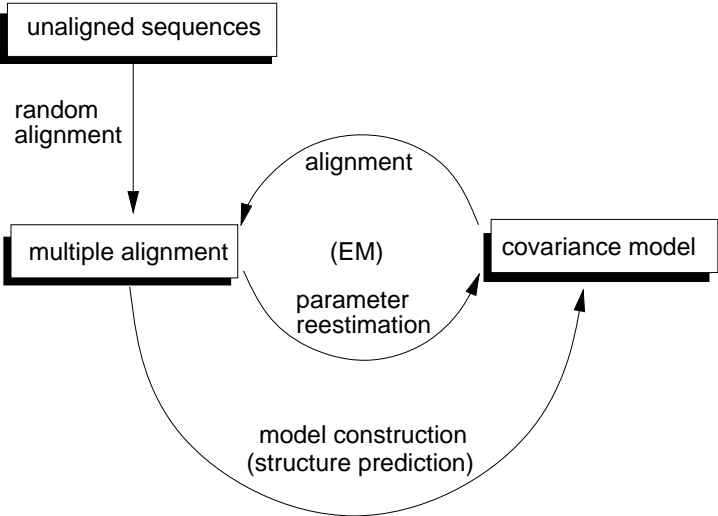
Dataset	Avg. id	Min id	Max id	ClustalV accuracy	1° info (bits)	2° info (bits)
TEST	.402	.144	1.00	64%	43.7	30.0-32.3
SIM100	.396	.131	.986	54%	39.7	30.5-32.7
SIM65	.362	.111	.685	37%	31.8	28.6-30.7

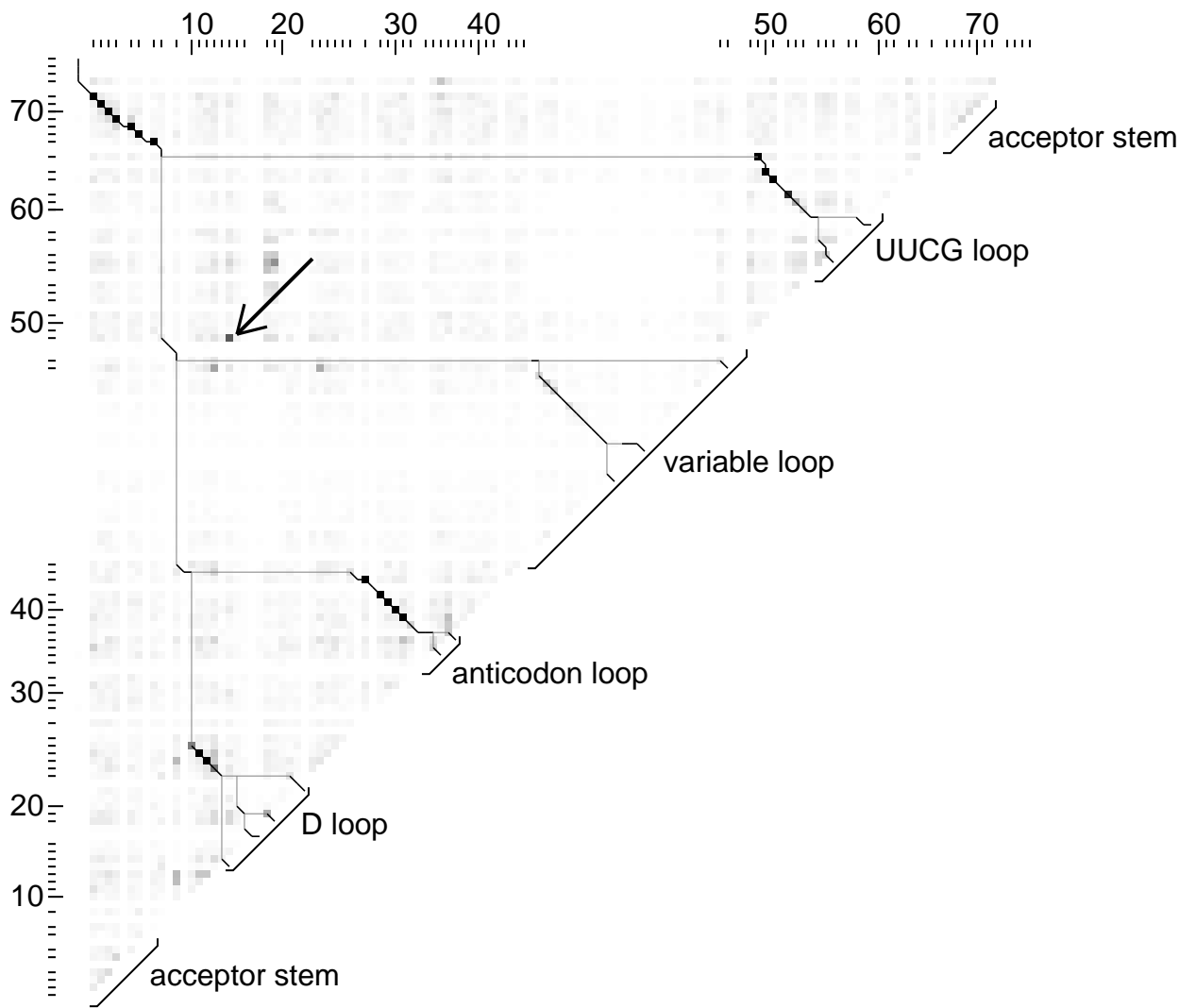
Table 1: Statistics of the training and test sets of 100 tRNA sequences each. The average identity in an alignment is the average pairwise identity of all aligned symbol pairs, with gap/symbol alignments counted as mismatches. Primary sequence information content is calculated according to [48]. Calculating pairwise mutual information content is an NP-complete problem of finding an optimum partition of columns into pairs. A lower bound is calculated by using the model construction procedure to find an optimal partition subject to a non-pseudoknotting restriction. An upper bound is calculated as sum of the single best pairwise covariation for each position, divided by two; this includes all pairwise tertiary interactions but overcounts because it does not guarantee a disjoint set of pairs. For the meaning of multiple alignment accuracy of ClustalV, see the text.

Model	training set	iterations	score (bits)	alignment accuracy
A1415	all sequences (aligned)	3	58.7	95%
A100	SIM100 (aligned)	3	57.3	94%
A65	SIM65 (aligned)	3	46.7	93%
U100	SIM100 (degapped)	23	56.7	90%
U65	SIM65 (degapped)	29	47.2	91%

Table 2: Training and multiple alignment results from models trained from the trusted alignments (A models) and models trained from no prior knowledge of tRNA (U models).







Trusted:

```
DF6280 GCGGAUUUAGCUCAGUU GGG AGAGCGCCAGACUGAAG AUCUGGAG GUCCUGUGUUCGAUCCACAGAAUUCGCACCA
DF6280G GCGGAUUUAGCUCAGUU GGG AGAGCGCCAGACUGAAGAAUACUUCGGUCAAGUUAUCUGGAG GUCCUGUGUUCGAUCCACAGAAUUCGCA
DD6280 UCCGUGAUAGUUUAAU GGUUCAGAAUGGGCGCUUGUCG CGUGCCAG A UCGGGGUCAAUUCGCCGUCGCGGAGCCA
DX1661 CGCGGGGUGGAGCAGCCUGGU AGCUCGUCGGGCUCAUA ACCCGAAG GUCGUCGGUCAAUUCGCCGCCCGCAACCA
DS6280 GGCAACUUGGCCGAGU GGUUAAGGCGAAAGAUUAGAA AUCUUUU GGGCUUUGCCCG CGCAGGUUCGAGUCCUGCAGUUGUCGCCA
```

U100:

```
DF6280 GCGGAUUUAGCUCAG UUGGGAGAGGCCAGACU GA AG AUCUGGA GGUCUGUGUUCGAUCCACAGAAUUCGCACca
DF6280G GCGGAUUUAGCUCAG UUGGGAGAGGCCAGACUgaagaaauacuUCgguCAaguuAUCUGGA GGUCUGUGUUCGAUCCACAGAAUUCGCA
DD6280 UCCGUGAUAGUUUAA UGGUCAGAAUGGGCGCUU GU CG CGUGCCA GAU CCGGGUCAAUUCGCCGUCGCGGAGcca
DX1661 CGCGGGGUGGAGCAGcCUGGUAGCUCGUCGGGCU CA UA ACCCGAA GGUCGUCGGUCAAUUCGCCGCCCGCAACca
DS6280 GGCAACUUGGCCGAG UGGUUAAGGCGAAAGAUU AG AA AUCUUUUgggcuuugcccG CGCAGGUUCGAGUCCUGCAGUUGUCGcca
```

ClustalV:

```
DF6280 GCGGAUUUAGCUCAGUUGGGAGAGGCCAGACUGAAGA UCUGGAGGUCCUGUGUUCGAUCCACAGAAUUCGCACCA
DF6280G GCGGAUUUAGCUCAGUUGGGAGAGGCCAGACUGAAGAAUACUUCGGUCAAGUUAUCUGGAGGUCCUGUGUUCGAUCCACAGAAUUCGCA
DD6280 UCCGUGAUAGUUUAAU G GUCAGAAUGG GCG CUUG UCGCUGGCC AGAUCGG GGUCAAUUCGCCGUCGCGGAGCCA
DX1661 CGCGGGGUGGAGCAGC CUGGUAGCUCGUCGGG CUCA UAACCCGA AGGUCGUCGGUCAAUUCGCCGCCCGCAACCA
DS6280 GGCAACUUGGCCGAGUGGUUAAGGCGAAAGAUU AGAAAUCUUUUGGGC UUUGCCCG CGCAGGUUCGAGUCCUGCAGUUGUCGCCA
```

number of hits

