M_{ij} is maximal if i and j individually appear completely random ($f_i = f_j = 0.25$), but i and j are perfectly correlated, for instance in a Watson–Crick base pair.

Intuitively, M_{ij} tells us how much information we get about the identity of the residue in one position if we are told the identity of the residue in the other position. In the case of a base pair with no sequence constraints, we get 2 bits of information: for instance, if we are told that i is a G, our uncertainty about j collapses from four possibilities to just one (C) so we gain 2 bits of information. If i and j are uncorrelated, the mutual information is zero. If either i or j are highly conserved positions, we also get little or no mutual information: if a position does not vary, we do not learn anything more about it by knowing the identity of its partner.

Figure 10.6 shows a contour plot of M_{ij} values calculated from a multiple alignment of 1415 tRNA sequences. The four base-paired stems of the cloverleaf structure are readily apparent. The D and TψCG stems, which are relatively highly conserved in primary sequence, are somewhat less apparent than the anticodon and acceptor stems which are extremely variable in primary sequence.

Exercise

10.1 The mutual information calculation in (10.1) requires counting frequencies of all sixteen different base pairs. This has the advantage that it makes no assumptions about Watson–Crick base pairing, so mutual information can be detected between covarying non-canonical pairs like A-A and G-G pairs. On the other hand, the calculation requires a large number of aligned sequences to obtain reasonable frequencies for sixteen possibilities. Write down an alternative information theoretic measure of base-pairing correlation that considers only two classes of i, j identities instead of all sixteen: Watson–Crick and G-U pairs grouped in one class, and all other pairs grouped in the other. Compare the properties of this calculation to the M_{ij} calculation both for small numbers of sequences and in the limit of infinite data.

10.2 RNA secondary structure prediction

Suppose we wish to predict the secondary structure of a single RNA. Many plausible secondary structures can be drawn for a sequence. The number increases exponentially with sequence length. An RNA only 200 bases long has over 10^{50} possible base-paired structures. We must distinguish the biologically

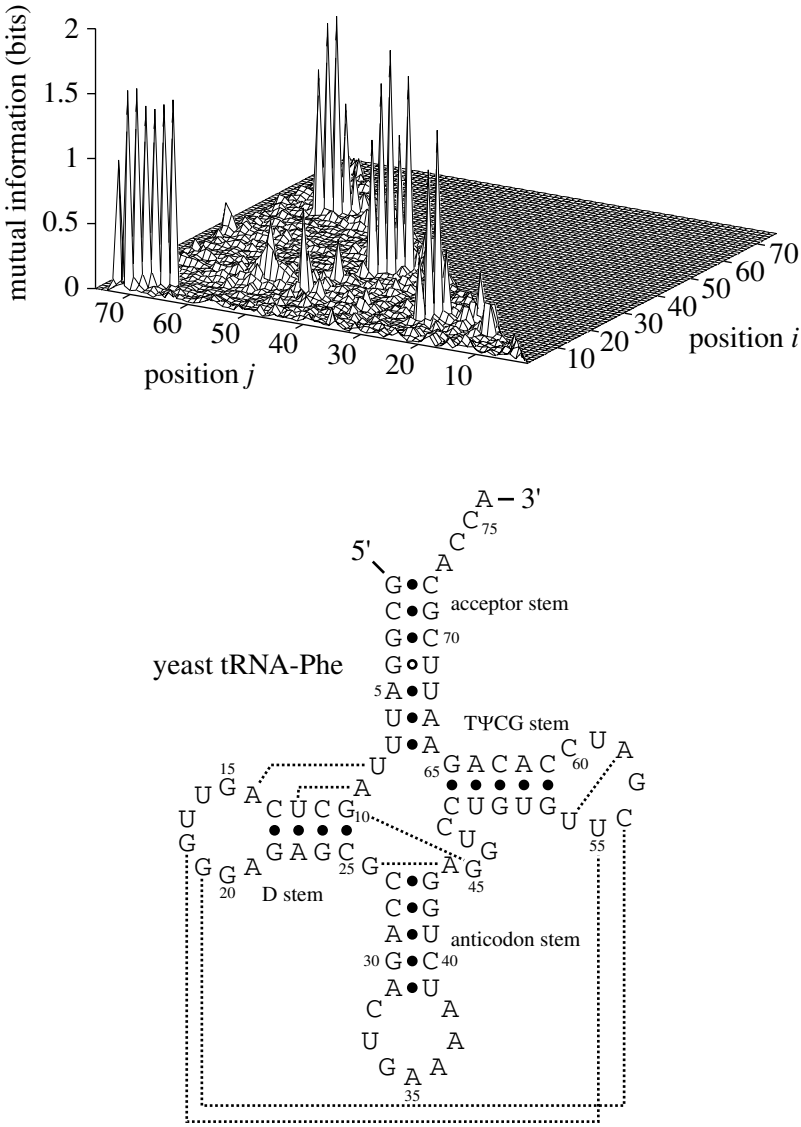

Figure 10.6 *A mutual information plot of a tRNA alignment (top) shows four strong diagonals of covarying positions, corresponding to the four stems of the tRNA cloverleaf structure (bottom; the secondary structure of yeast phenylalanine tRNA is shown). Dashed lines indicate some of the additional tertiary contacts observed in the yeast tRNA-Phe crystal structure. Some of these tertiary contacts produce correlated pairs which can be seen weakly in the mutual information plot.*

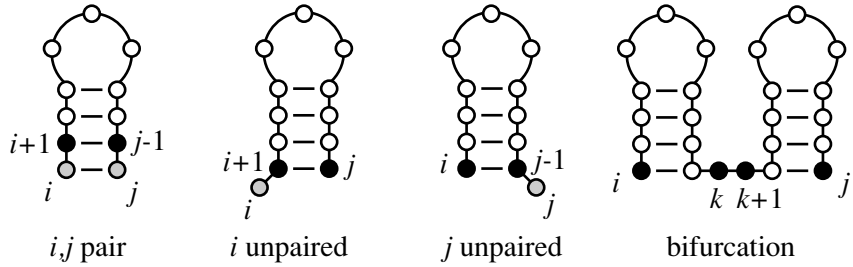

Figure 10.7 *The Nussinov algorithm looks at four ways in which the best RNA structure for a subsequence i, j can be made by adding i and/or j onto already calculated optimal structures for smaller subsequences. Pseudoknots are not considered.*

correct structure from all the incorrect structures. We need both a function that assigns the correct structure the highest score, and an algorithm for evaluating the scores of all possible structures.

Base pair maximisation and the Nussinov folding algorithm

One approach might be to find the structure with the most base pairs. Nussinov introduced an efficient dynamic programming algorithm for this problem [Nussinov *et al*. 1978]. Although this criterion is too simplistic to give accurate structure predictions, the example is instructive because the mechanics of the Nussinov algorithm are the same as those of the more sophisticated energy minimisation folding algorithms and of probabilistic SCFG-based algorithms.

The Nussinov calculation is recursive. It calculates the best structure for small subsequences, and works its way outwards to larger and larger subsequences. The key idea of the recursive calculation is that there are only four possible ways of getting the best structure for i, j from the best structures of the smaller subsequences (Figure 10.7):

(1) add unpaired position i onto best structure for subsequence $i+1, j$;
(2) add unpaired position j onto best structure for subsequence $i, j-1$;
(3) add i, j pair onto best structure found for subsequence $i+1, j-1$;
(4) combine two optimal substructures i, k and $k+1, j$.

More formally, the Nussinov RNA folding algorithm is as follows. We are given a sequence x of length L with symbols $x_1, \ldots, x_L$. Let $\delta(i, j) = 1$ if x_i and x_j are a complementary base pair; else $\delta(i, j) = 0$. We will recursively calculate scores $\gamma(i, j)$ which are the maximal number of base pairs that can be formed for subsequence $x_i, \ldots, x_j$.

Algorithm: Nussinov RNA folding, fill stage

Initialisation:

$$\begin{aligned} \gamma(i,i-1) &= 0 \qquad \text{for } i = 2 \text{ to } L; \\ \gamma(i,i) &= 0 \qquad \text{for } i = 1 \text{ to } L. \end{aligned}$$

Recursion: starting with all subsequences of length 2, to length L:

$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j), \\ \gamma(i,j-1), \\ \gamma(i+1,j-1)+\delta(i,j), \\ \max_{i<k<j}\left[\gamma(i,k)+\gamma(k+1,j)\right]. \end{cases}$$ ◁

Figure 10.8 shows an example of a Nussinov matrix fill in operation.

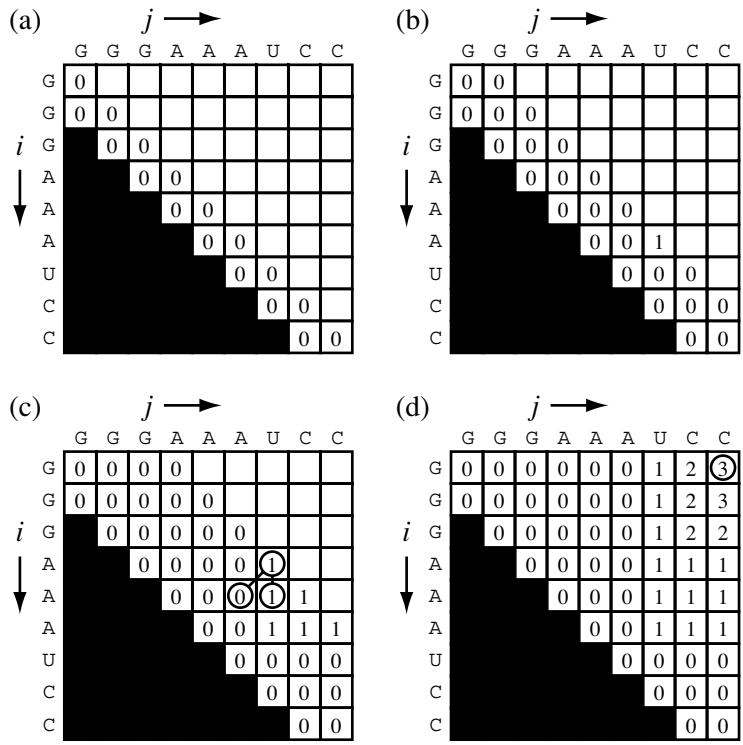

Figure 10.8 *The matrix fill stage of the Nussinov folding algorithm is shown for an example sequence* GGGAAAUCC. *(a) The initialised half-diagonal matrix. (b) The matrix after scores for subsequences of length two have been calculated. (c) An example of two different optimal substructures for the same subsequence. For the subsequence* AAAU, *either the* A *at* i *and the* U *at* j *can be paired (diagonal path) or* i *can be added to a substructure that already pairs the* A *at* $i+1$ *to the* U *at* j *(vertical path). (d) The final matrix. The value in the upper right indicates that the maximally paired structure has three base pairs.*

The value of $\gamma(1,L)$ is the number of base pairs in the maximally base-paired structure. There are often a number of alternative structures with the same number of base pairs. To find one of these maximally base-paired structures, we trace back through the values we calculated in the dynamic programming matrix, beginning from $\gamma(1,L)$. In pseudocode, the traceback algorithm is:

Algorithm: Nussinov RNA folding, traceback stage

Initialisation: Push $(1,L)$ onto stack.
Recursion: Repeat until stack is empty:

- pop (i,j).
- if $i >= j$ continue;
  else if $\gamma(i+1,j) = \gamma(i,j)$ push $(i+1,j)$;
  else if $\gamma(i,j-1) = \gamma(i,j)$ push $(i,j-1)$;
  else if $\gamma(i+1,j-1)+\delta_{i,j} = \gamma(i,j)$:
    - record i,j base pair.
    - push $(i+1,j-1)$.
  else for $k = i+1$ to $j-1$: if $\gamma(i,k)+\gamma(k+1,j) = \gamma(i,j)$:
    - push $(k+1,j)$.
    - push (i,k).
    - break. ◁

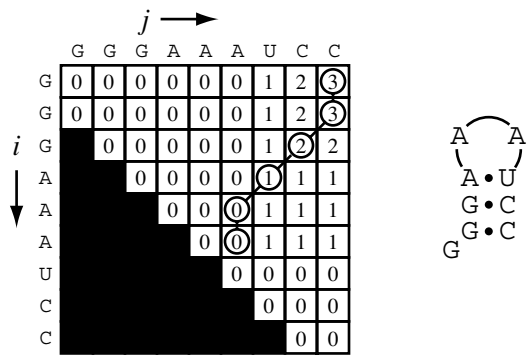

Figure 10.9 *The traceback stage of the Nussinov folding algorithm is shown for the filled matrix from Figure 10.8. An optimal traceback path is indicated with circles. The optimal structure corresponding to this path is shown at right.*

The traceback is linear in time and memory. The fill step is the limiting step as it is $O(L^2)$ in memory and $O(L^3)$ in time. An example traceback is shown in Figure 10.9. The traceback in Figure 10.9 is unbranched, so the need for the pushdown stack in the traceback algorithm is not apparent. The pushdown stack

becomes important when bifurcated structures are traced back. The stack remembers one side of the the bifurcation while the other side is traced back, reminiscent of the push-down automata in Chapter 9.

Exercises

10.2 The traceback algorithm given above does not actually produce the structure shown in Figure 10.9. What alternative optimal structure containing three base pairs does it recover instead? Are there other optimal structures? Modify the traceback algorithm so it finds a different optimal structure.

10.3 As we have given it, the Nussinov algorithm can produce nonsensical 'base pairs' between adjacent complementary residues, with a physically improbable loop length of zero (for example, you should have seen one such structure in the preceding exercise) Modify the Nussinov folding algorithm so that hairpin loops must have a minimum length of h. Give the new recursion equations for the fill and traceback.

10.4 Show that the Nussinov folding algorithm can be trivially extended to find a maximally *scoring* structure where a base pair between residues a and b gets a score $s(a,b)$. (For instance, we might set $s(\mathtt{G},\mathtt{C}) = 3$ and $s(\mathtt{A},\mathtt{U}) = 2$ to better reflect the increased thermodynamic stability of GC pairs.)

An SCFG version of the Nussinov algorithm

The Nussinov algorithm is fundamentally similar to the SCFG algorithms in Chapter 9. As an example of how SCFGs apply to RNA secondary structure analysis, consider the following production rules of a simple RNA folding SCFG:

$$\begin{array}{lcll} S & \rightarrow & aS \mid cS \mid gS \mid uS & (i \text{ unpaired}), \\ S & \rightarrow & Sa \mid Sc \mid Sg \mid Su & (j \text{ unpaired}), \\ S & \rightarrow & aSu \mid cSg \mid gSc \mid uSa & (i,j \text{ pair}), \\ S & \rightarrow & SS & (\text{bifurcation}), \\ S & \rightarrow & \epsilon & (\text{termination}). \end{array} \qquad (10.2)$$

The SCFG has a single nonterminal S and 14 production rules with associated probability parameters. For now, assume that the probability parameters are known. The maximum probability parse of a sequence with this SCFG is an assignment of sequence positions to productions. Because the productions correspond to secondary structure elements (base pairs and single-stranded bases), the maximum probability parse is equivalent to the maximum probability secondary structure. If base pair productions have relatively high probability, the SCFG will favour parses which tend to maximise the number of base pairs in the structure.

Although the production rules for the SCFG are not in Chomsky normal form, a CYK parsing algorithm is readily written that finds the maximum probability secondary structure. Alternatively, we could convert the SCFG to Chomsky normal form and apply the algorithms in Chapter 9. Although the Chomsky normal form approach is attractive in its generality, specific algorithms for specific SCFGs are typically more efficient. The adapted CYK algorithm is as follows. Let the probability parameters of the SCFG productions be denoted by $p(aS)$, $p(aSu)$, etc.

Algorithm: CYK for Nussinov-style RNA SCFG

Initialisation:

$$\begin{aligned} \gamma(i,i-1) &= -\infty \qquad \text{for } i = 2 \text{ to } L; \\ \gamma(i,i) &= \max \begin{cases} \log p(x_i S) \\ \log p(S x_i) \end{cases} \qquad \text{for } i = 1 \text{ to } L. \end{aligned}$$

Recursion: for $i = L-1$ down to 1, $j = i+1$ to L:

$$\gamma(i,j) = \max \begin{cases} \gamma(i+1,j) + \log p(x_i S); \\ \gamma(i,j-1) + \log p(S x_j); \\ \gamma(i+1,j-1) + \log p(x_i S x_j); \\ \max_{i<k<j} \gamma(i,k) + \gamma(k+1,j) + \log p(SS). \end{cases} \qquad \triangleleft$$

When this is done, $\gamma(1,L)$ is the log likelihood $\log P(x,\hat{\pi}|\theta)$ of the optimal structure $\hat{\pi}$ given the SCFG model θ. The traceback to find the structure corresponding to that best score is either performed analogously to the traceback in the Nussinov algorithm, or by keeping additional traceback pointers in the fill stage analogous to the CYK algorithm description in Chapter 9.

The principal difference between this and the original Nussinov algorithm is that the SCFG description is a probabilistic model. We gain access to several well-principled options for optimising the parameters of the model. We can set the SCFG's parameters by subjective estimation of the relevant probabilities, or by estimating parameters by counting state transitions in known RNA structures and converting the counts to probabilities. We can even learn probabilities from example RNAs of *unknown* structure using expectation maximisation (EM) and inside–outside training to iteratively infer both the structures and the parameters (i.e. the structures are the hidden data in the EM algorithm). Once we have written down the SCFG as a full probabilistic model of the RNA folding problem, we can 'turn the crank', applying all the probabilistic machinery we have learned in previous chapters almost by rote.

Like the Nussinov algorithm, this small SCFG is a good starting example but it is too simple to be an accurate RNA folder. It does not consider important structural features like preferences for certain loop lengths nor preferences for

certain nearest neighbours in the structure caused by stacking interactions between neighbouring base pairs in a stem.

Exercises

10.5 Write down a traceback algorithm for determining the best RNA secondary structure after the above algorithm has completed.

10.6 Devise an SCFG which uses different nonterminals to model bulge loops, hairpin loops, multifurcation loops and single strands.

Energy minimisation and the Zuker folding algorithm

RNA folding is dictated by biophysics rather than by counting and maximising the number of base pairs. The most sophisticated secondary structure prediction method for single RNAs is the Zuker algorithm, an energy minimisation algorithm which assumes that the correct structure is the one with the lowest equilibrium free energy (ΔG) [Zuker & Stiegler 1981; Zuker 1989a].

The ΔG of an RNA secondary structure is approximated as the sum of individual contributions from loops, base pairs and other secondary structure elements. An important difference from the simpler Nussinov calculation is that the energies of stems are calculated by adding *stacking* contributions for the interface between neighbouring base pairs instead of individual contributions for each pair. In other words, the energy of a stem of n base pairs is the sum of $n-1$ base stacking terms instead of n base pair terms. This produces a better fit to experimentally observed ΔG values for RNA structures but it complicates the dynamic programming algorithm. Tables of ΔG parameters for RNA structure prediction have been fitted to the results of experimental thermodynamic studies of small model RNAs [Freier *et al*. 1986; Turner *et al*. 1987]. They include parameters for stacking, hairpin loop lengths, bulge loop lengths, interior loop lengths, multi-branch loop lengths, single dangling nucleotides and terminal mismatches on stems.

An example of the prediction of the ΔG of an RNA structure is given in Figure 10.10. Single base bulges are assumed not to disrupt stacking in the stem, so a stacking term is included in the example in the figure. Longer bulges, which are assumed to disrupt stacking, get no added stacking term. The hairpin loop energy is the sum of two terms: a loop destabilisation energy dependent only on the loop length, and a terminal mismatch energy dependent on the closing base pair and the first and last bases of the stem. The energies used in Figure 10.10 are from the older 'Freier rules' [Freier *et al*. 1986] at 37°C.[1]

The minimum energy structure can be calculated recursively by a dynamic programming algorithm (assuming no pseudoknots), very similar to how the

[1] Currently the most up-to-date parameters are available on the Web from `http://www.bioinfo.rpi.edu/~zukerm/rna/energy/`.

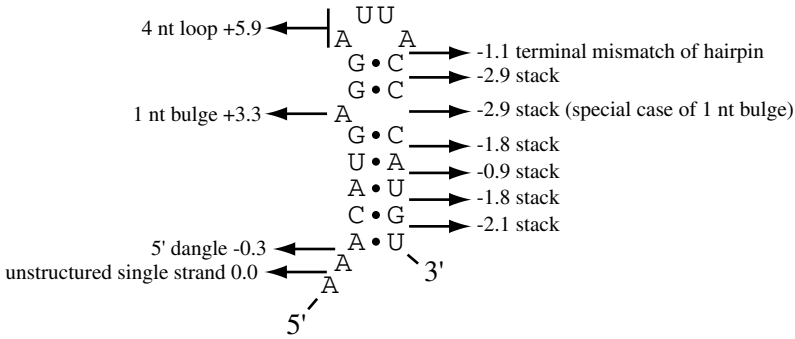

Figure 10.10 *An example ΔG calculation for an RNA stem loop (the wild type R17 coat protein binding site).*

maximum base-paired structure was calculated above. The principal difference is that because of the stacking parameters, two matrices (called V and W) are kept instead of one. $W(i,j)$ is the energy of the best structure on i,j. $V(i,j)$ is the energy of the best structure on i,j given that i,j are paired. The algorithm can then keep track of stacking interactions by adding new base pairs only onto the V matrix. Conceptually this two-state calculation is very similar to the use of extra insert states in pairwise dynamic programming alignment with affine gap costs (Chapter 2) to keep track of insert extensions. For a complete description of the Zuker algorithm, see Zuker & Stiegler [1981].

We could write down a SCFG that followed similar rules. The simplest stacking production rule would be, for instance, $cVg \rightarrow cgVcg$ for producing a GC pair in a stem after (stacked on) a CG, using V as a base pair generating nonterminal (as in the Zuker V matrix). With the CG terminals on the left as context for the production of the GC, this is technically a context-sensitive production, so we can't use such rules as the basis for a SCFG. However, we can convert to context-free productions by using four different nonterminals $V^{au}, V^{cg}, V^{gc}, V^{ua}$, and using right-hand sides of the form $\rightarrow gV^{gc}c$ to produce a G-C pair, for instance – the nonterminal identity V^{gc} 'remembers' that a G-C pair was just generated. (In other words, all we are doing is making the model a higher order Markov process.) The probability of a production $V^{cg} \rightarrow gV^{gc}c$, for instance, would be the probability of a C-G pair stacked on a G-C pair.[2] Other details of the Zuker algorithm and its two matrices V and W could be incorporated similarly into an analogous full probabilistic model with two nonterminals V and W (expanded for nearest neighbour context). CYK and inside–outside algorithms for an SCFG

[2] Since only one nonterminal is possible for a given x_i, x_j pair and the other three have zero probability, the four nonterminals behave as one for the purposes of memory and time complexity in parsing algorithms.

version of the Zuker algorithm have the same algorithmic complexity as the Zuker algorithm itself.

Suboptimal RNA folding

The original Zuker algorithm finds only the optimal structure. The biologically correct structure is often not the calculated optimal structure, but rather a structure within a few percent (i.e. within the error bars) of the calculated minimum energy. It was a significant advance when an efficient suboptimal folding algorithm was introduced. The Zuker *suboptimal* folding algorithm [Zuker 1989b] is similar to running the CYK algorithm in both the inside and outside directions. One matrix (exactly the CYK algorithm) finds the ΔG of the best structure for all subsequences i, j with i, j paired, and a second matrix (effectively an outside CYK algorithm) finds the best structure for the sequence with i, j paired and the subsequence $i+1, j-1$ excluded.[3] The sum of the two numbers for a given i, j is the ΔG of the optimal structure that uses the pair i, j. The suboptimal folding algorithm then samples a base pair i, j 'randomly' according to its ΔG, then traces back in both the inside and outside matrices to find the optimal structure that uses that base pair. (It is therefore more correct to say that the algorithm samples *one base pair* suboptimally. The rest of the structure is the optimal structure given that base pair.)

SCFG versions of RNA folding algorithms can also sample structures according to their likelihood by a probabilistic traceback of the inside matrix, analogous to the way in which suboptimal profile HMM alignments were sampled from a forward matrix in Chapter 6.

Base pair confidence estimates

Partition function calculations for calculating the probabilities of particular base pairs or structures were introduced for energy minimisation folding algorithms by McCaskill [1990]. The McCaskill algorithm converts ΔGs to probabilities using the Gibbs–Boltzmann equation and sums probabilities of all structures instead of choosing the single minimum energy structure. The sum of the probabilities of all structures containing a base pair i, j divided by the sum over all structures is interpreted as a confidence estimate in the pair i, j.

From the SCFG viewpoint, the McCaskill algorithm is fundamentally an inside–outside algorithm, compared to the Zuker algorithm which is fundamentally a

[3] Zuker actually doubles the sequence, treats it as circular, and calculates the energy of the best structure on $j, \ldots, L/1, \ldots, i$. For circular RNAs, this gives the same result as the outside algorithm. For linear RNAs, the Zuker algorithm must handle the non-existent junction between the 3′ and 5′ end as a special case. The outside algorithm might be less complicated to implement.

CYK algorithm. The estimation of base pair confidences for an SCFG is conceptually similar to the estimation of pairwise alignment confidences that we described for pair HMMs in Chapter 4.

Exercises

10.7 Write down the inside algorithm, outside algorithm, and inside–outside re-estimation equations for the Nussinov-style RNA folding SCFG in equation (10.2).

10.8 By analogy to profile HMM suboptimal alignment sampling, give an algorithm for sampling structures probabilistically from your inside matrix.

10.9 Show how to use your inside and outside variables to calculate the probability that positions i, j are base-paired, summed over all structures. The functional form of the answer will be analogous to your inside–outside re-estimation equations.

10.3 Covariance models: SCFG-based RNA profiles

Suppose we have a family of related RNAs – transfer RNAs or group I catalytic introns, perhaps – which share a common consensus secondary structure as well as some primary sequence motifs, and we want to search a sequence database for homologous RNAs. In Chapter 5, we used HMM-based profiles to model the consensus of protein and DNA sequence families, but we showed in Chapter 9 that HMMs are primary structure models that cannot deal effectively with RNA secondary structure constraints. In this section, we describe SCFG-based RNA structure profiles called 'covariance models' (CMs) which are the SCFG analogue of profile HMMs. Whereas profile HMMs specify a repetitive linear HMM architecture well suited for modelling multiple sequence alignments, CMs specify a repetitive tree-like SCFG architecture suited for modelling consensus RNA secondary structures.

Although we follow here the 'covariance model' approach developed in Eddy & Durbin [1994], these same general ideas and algorithms are shared by comparable SCFG-based RNA models independently developed at the same time by Sakakibara and coworkers [1994].

CMs are detailed and fairly complex probabilistic models. We first set the stage by looking in an intuitive way at more simple models of small RNA alignments.

SCFG models of ungapped RNA alignments

Figure 10.11 shows an example RNA consensus structure and an ungapped multiple alignment of an RNA family that fit the consensus. To describe this