SGN-6156, Lecture 5
Biological sequence analysis

Harri Lähdesmäki, harri.lahdesmaki@tut.fi

Department of Signal Processing,

Tampere University of Technology

15.04.2008

Pairwise vs. family alignment

- This lecture is based on Section 5 in (Durbin et al., 1998)
- Previous methods focus on aligning sequence pairs (x, y)
- Many functional biological sequences come in families
- A straightforward approach: align a sequence x with all sequences y in a family $\mathcal{Y}$
- Pairwise comparisons can miss distantly related sequences, but detection sensitivity can be improved using conserved features of a family
- An example in Figure 5.1 (Durbin et al., 1998)
- A probabilistic family alignment using profile HMMs
- Assume we are given an alignment of multiple sequences

Ungapped score matrix

- In Figure 5.1 ungappad/gapped regions are relatively well aligned
- Define a score for an ungapped region as

$$P(x|M) = \prod_{i=1}^{L} e_i(x_i)$$

  where $e_i(x_i)$ is the probability of seeing nucleic/amino acid x_i in position i

- Compare this with the random model, i.e.,

$$S = \log P(x|M)/P(x|R) = \sum_{i=1}^{L} \log \frac{e_i(x_i)}{q_{x_i}}$$

- $\log \frac{e_i(x_i)}{q_{x_i}}$ terms defines a position specific score matrix (PSSM) which does not allow gaps

Profile HMMs

- PSSM is a special type of HMM: sequence of “match states” M_i with emission probabilities $e_{M_i}(a)$ and deterministic transitions between them (see Figures on pages 103–104)
- Some positions are more prone to gaps than others
- Insertions can be anywhere in the sequence: move from match state M_i to insertion state I_i and back to M_{i+1}
  - Score penalty of an insertion is equal to the sum of $\log$ transition probabilities ($a_{M_i I_i}$, $a_{I_i I_i}$ and $a_{I_i M_{i+1}}$)
- Deletions anywhere in the sequence: move from match state M_i to another match state M_j, $j > i + 1$, via salient states D_{i+1}, D_{i+2}, etc.
  - Score penalty of a deletion is equal to the sum of $\log$ transition probabilities ($a_{M_i D_{i+1}}$, $a_{D_{i+1} D_{i+2}}$, $\ldots$, $a_{D_{j-1} D_j}$, $a_{D_j M_{j+1}}$)

- Profile HMM is obtained by putting together the three parts: PSSM, insertions and deletions
- Profile HMMs can be seen as a generalization of pair HMMs. Notice that the structure of profile HMM is in a sense repetitive compared to that of pair HMM
- Thus, practically the same algorithms as in the case of pair HMMs can be applied
- note that transition probabilities $a_{M_i D_{i+1}}$ can be different from $a_{M_j D_{j+1}}$ for $i \neq j$ (position specificity)
- See Figure 5.4 (Durbin et al., 1998)

Parameters of profile HMMs

- Profile HMM can be thought of as a stochastic process (“random number generator”) that generates sequences from a family
- Members of a particular family should be assigned a high probability
- The structure of a profile HMM can be constructed based on the multiple aligned (which we assume is available)
- State transition probabilities can be estimated using ML principle (again assuming a multiple alignment is given)

Profile HMMs and searching

- Use profile HMM to match/align a new/unannotated sequence x to a family
- Most probable alignment (Viterbi algorithm)
- The probability of x, summed over all alignments (forward algorithm)
- Instead of pure probabilities, log-odds are used (length dependency)
- Let $V_j^M(i)$ denote the score of the best path that matches $x_1, \ldots, x_i$ to the profile HMM until state M_j and ending with symbol x_i ($V_j^I(i)$ and $V_j^D(i)$ similarly)

Viterbi for profile HMMs

- Viterbi recursions:

$$V_j^M(i) = \log \frac{e_{M_j}(x_i)}{q_{x_i}} + \max \begin{cases} V_{j-1}^M(i-1) + \log a_{M_{j-1}M_j} \\ V_{j-1}^I(i-1) + \log a_{I_{j-1}M_j} \\ V_{j-1}^D(i-1) + \log a_{D_{j-1}M_j} \end{cases}$$

$$V_j^I(i) = \log \frac{e_{I_j}(x_i)}{q_{x_i}} + \max \begin{cases} V_j^M(i-1) + \log a_{M_j I_j} \\ V_j^I(i-1) + \log a_{I_j I_j} \\ V_j^D(i-1) + \log a_{D_{j-1} I_j} \end{cases}$$

$$V_j^D(i) = \max \begin{cases} V_{j-1}^M(i) + \log a_{M_{j-1}D_j} \\ V_{j-1}^I(i) + \log a_{I_{j-1}D_j} \\ V_{j-1}^D(i) + \log a_{D_{j-1}D_j} \end{cases}$$

Forward algorithm for profile HMMs

- Let $F_j^M(i)$ denote the full score of the subsequence $x_1, \ldots, x_i$ to the profile HMM until state M_j and ending with symbol x_i ($V_j^I(i)$ and $V_j^D(i)$ similarly)
- The forward algorithm is practically the same as the Viterbi except that $\max$ is replaced with summation

Profile HMM example

- See pages 111–112/Figures 5.5–5.6 (Durbin et al., 1998)
- A profile HMM for local alignment (see page 113 (Durbin et al., 1998))

Multiple sequence alignment

- The material follows Section 6 in (Durbin et al., 1998)
- Previously we have considered both pairwise alignments or family alignments using profile HMMs (assuming a multiple alignment was given)
- Good multiple alignments can be constructed manually by experts but that is a slow process
- Probabilistic multiple alignments can be constructed computationally
- Briefly, similar/homologous residues in sequences are aligned in columns
- It is impossible in general to construct a single meaningful best alignment

A score for multiple alignments

- Multiple alignments make use of the observation that some part are more conserved than others, see Figure 6.1 (Durbin et al. 1998)
- Notation:
  - m is the multiple alignment (matrix) and m_i^j defines the symbol for sequence j in column i
  - The ith column is $m_i = (m_i^1, \ldots, m_i^N)^T$
  - c_{ia} is the number of times symbol a occurs in column i (for all a)
- A simplifying assumption: columns m_i of a multiple alignment m are independent

$$S(m) = G + \sum_i S(m_i)$$

$S(m_i)$ is the score for a column and G adds a penalty for gaps

Minimum entropy score

- If residues in a column are independent then the probability of a column can be written as

$$P(m_i) = \prod_{j=1}^{N} p_{im_i^j} = \prod_a p_{ia}^{c_{ia}}$$

where p_{ia} is the probability of observing symbol a in column i, and an entropy score can be defined as

$$S(m_i) = -\log P(m_i) = \sum_a c_{ia} \log p_{ia}$$

- Probabilities for residues p_{ia} can be estimated from the counts c_{ia} using ML principle

Sum of pairs score

- Columns can be scored by sum of pairs using a substitution matrix s (e.g. BLOSUM or PAM)
- A column score can be written as

$$S(m_i) = \sum_{k<l} s(m_i^k, m_i^l)$$

- Linear gap scores can be handled using a similar formulation $s(a, \text{‘gap’})$, $s(\text{‘gap’}, a)$, and $s(\text{‘gap’}, \text{‘gap’})$

Multidimensional dynamic programming

- Pairwise dynamic programming alignment can be generalized to multiple sequences
- Assume statistically independent columns and linear gap penalty
- Define $\alpha_{i_1,i_2,\ldots,i_N}$ to be the maximum alignment score for subsequences (and ending with) $(x_1^1,\ldots,x_{i_1}^1)$, $(x_1^2,\ldots,x_{i_2}^2)$, ..., $(x_N^1,\ldots,x_{i_N}^N)$

- Multidimensional dynamic programming recursions: $2^N - 1$ cases

$$\alpha_{i_1,i_2,\ldots,i_N} = \max \begin{cases} \alpha_{i_1-1,i_2-1,\ldots,i_N-1} & + \quad S(x^1_{i_1}, x^2_{i_2}, \ldots, x^N_{i_N}) \\ \alpha_{i_1,i_2-1,\ldots,i_N-1} & + \quad S(\text{`gap'}, x^2_{i_2}, \ldots, x^N_{i_N}) \\ \alpha_{i_1-1,i_2,\ldots,i_N-1} & + \quad S(x^1_{i_1}, \text{`gap'}, \ldots, x^N_{i_N}) \\ & \vdots \\ \alpha_{i_1-1,i_2-1,\ldots,i_N} & + \quad S(x^1_{i_1}, x^2_{i_2}, \ldots, \text{`gap'}) \\ \alpha_{i_1,i_2,i_3-1\ldots,i_N-1} & + \quad S(\text{`gap'}, \text{`gap'}, \ldots, x^N_{i_N}) \\ & \vdots \\ \alpha_{i_1,i_2-1,\ldots,i_{N-1}-1,i_N} & + \quad S(\text{`gap'}, x^2_{i_2}, \text{`gap'}) \\ & \vdots \end{cases}$$

- Dynamic programming matrix size is $L_1 L_2 \ldots L_N$

- Each element requires maximization over the $2^N - 1$ different cases
- Assuming all sequences have approximately the same length $L \approx L_i$, then time complexity is $O(2^N L^N)$
- An alternative is to define the score to be the sum of pairwise alignment. In that case, MSA is an efficient algorithm for multiple alignment
- A number of heuristic methods have been developed

Progressive multiple alignment methods

- Progressive alignment methods are heuristic, but perhaps the most commonly used in practise. A general method is as follows
  - Align two sequence using a pairwise method
  - Align a third sequence to the previous alignment/profile
  - Continue this process for all the remaining sequences
- Different variants have been proposed
  - The order in which sequences are aligned
  - Whether sequences are aligned with the single growing alignment, or subfamily alignments are first constructed and the families are then aligned
  - The methods to compute pairwise and family alignments
- Align the most similar sequences first

ClustalW algorithm

- ClustalW is a popular multiple alignment method
  - Construct a distance matrix from all $N(N-1)/2$ pairwise alignments
  - Construct a guide tree (phylogenetic tree) from the pairwsie distances using a clustering algorithm
  - Progressively align sequences/family in the order of decreasing distance
- ClustalW has a number of additional heuristics

Iterative refinement methods

- A problem with progressive alignment methods is that previously computed alignments are kept fixed
- Barton-Sternberg multiple alignment
  - Align the two most similar sequences (pairwise)
  - Align to the profile (of the two sequences) the most similar sequence. Repeat for all remaining sequences
  - Remove one sequence from the alignment/profile and re-align. Repeat for all sequences
  - Repeat the re-alignment step

Fully probabilistic multiple alignment

- Profile HMM training: simultaneous alignment and parameter estimation

References

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