BINF 730 – Lecture 3

Local and Global Alignment

Conditional Probability

- P(B) is the probability that event B occurs
- P(A and B) is the probability that both and A and B occur
- P(A or B) is the probability that either A or B or both occur
- P(A|B) is the probability that occurs given that B has already occurred
- Bayes' Rule

$$P(A|B) = P(A \text{ and } B)/P(B)$$

comes from

$$P(A \text{ and } B) = P(B)\, P(A|B)$$

Bayesian Statistics

- Suppose a gene A has two possible alleles A1 and A2 and another gene B has two possible states B1 and B2
- P(B) = P(B1) + P(B2) =1 and P(A) = P(A1) + P(A2) = 1
- Suppose that we know P(B1) = 0.3, then P(B2) = 1 – 0.3 = 0.7
- Suppose that we also know that P(A1|B1) = 0.8 and P(A2|B2) = 0.7.
- Since P(A1|B1) + P(A2|B1) = 1, P(A2|B1) = 1 – 0.8 = 0.2
- Since P(A1|B2) + P(A2|B2) = 1, P(A1|B2) = 1 – 0.7 = 0.2

- How do we find the joint probabilities of A1 and B1?

Bayesian Statistics

	A1	A2	
B1	0.24	0.06	0.3
B2	0.21	0.49	0.7
	0.45	0.55	1.0

- How do fill in this table?

- Use Bayesian Statistics
- P(A1 and B1) = P(B1)P(A1|B1) = 0.3 x 0.8 = 0.24
- P(A2 and B2) = P(B2)P(A2|B2) = 0.7 x 0.7 = 0.49
- P(A1 and B2) = P(B2)P(A1|B2) = 0.7 x 0.3 = 0.21
- P(A2 and B1) = P(B1)P(A2|B1) = 0.3 x 0.2 = 0.06

Local Alignment for Distantly Related Proteins

- The PAM matrices devised by Dayhoff works for closely related protein sequences.
- For proteins not closely related Dayhoff introduced an additional parameter t which represents the evolutionary time scale
- Hence

$$p_{AB}(t) = P(A, B \mid t)$$

$$p_{AB}(t) = P(A, B \mid t)$$

$P(B \mid A, t)$ is the probability that amino acid A is substituted by B within evolutionary distance t.

$$P(A, B \mid t) = P(B \mid A, t) P(A \mid t)$$

Assume that the amino acid distribution does not change during evolution. Then

$$P(A \mid t) = P(A) = q_A$$

$$P(A, B \mid t) = P(B \mid A, t) q_A$$

$$P(A,B \mid t) = P(B \mid A,t) q_A$$

$$P(B \mid A,t) = \frac{P(A,B \mid t)}{q_A}$$

Hence, P(B|A,t) can be estimated from the relative frequency of the pair (A,B) in the known alignment of two sequences s and s' with distance t and from the relative frequency q_A of the amino acid.

This allows us to generate a substitution matrix M' for all pairs of amino acids for a specific evolutionary distance t, using the matrix

$$M = (P(B \mid A,t))_{AB}$$

$$M' = (P(B \mid A,kt))_{AB}$$

$$M' = (P(B \mid A,t))_{AB} \times (P(B \mid A,t))_{AB} \times \cdots \times (P(B \mid A,t))_{AB} \quad k-times$$

$$M' = M^k$$

M is the PAM matrix (Point Accepted Mutation)

Evolutionary Time

- Two sequences s and s' have an evolutionary distance of 1 PAM unit if s was converted to s' by a series of accepted substitutions with an average of 1 accepted substitution per 100 amino acids
- Note that it is accepted substitution – backsubstitutions are ignored in this model.
- In other words a substitution matrix is 1 PAM unit if the expected number of substitutions in a typical protein is 1%.
- Hence $N=M^n$ is a n PAM matrix

Evolutionary Time

- For sequences whose distance is thought to be more than 100 PAM units distance, the PAM250 matrix is used.

Finding M

- It is not likely to find homologous sequences s and s' that are exactly 1 PAM distant.
- A scaling factor λ was chosen so that the matrix generated by

$$P(B \mid A,t) = \lambda \frac{n_{AB}(s,s')}{q_A} \quad for\ B \neq A$$

$$P(A \mid A,t) = 1 - \sum_{B \neq A} P(B \mid A,t)$$

is 1 PAM.

Caveats

- Since it is difficult to extrapolate to distantly related proteins from closely related units other substitution matrices have been proposed such as BLOSUM.
- BLOSUM is better for distantly related proteins.

Global Pairwise Sequence Alignment

- Distance Methods – how different are the sequences
- Alignment with Tandem Duplication - Uses the DSI model which is a different evolutionary model with duplications, substitutions and indels
- Similarity Methods - align many sequences with of different lengths and find optimal alignment of all sequences

Definitions

- Edit Operation
- Edit Distance
- Metric
- Alignment
- Alignment Distance

Edit Operation

Given an *alphabet* Σ with $- \notin \Sigma$, an *edit operation* is a pair

$$(x, y) \in (\Sigma \cup \{-\}) \times (\Sigma \cup \{-\})$$

We say that an edit operation is a

- Substitution if $x, y \in \Sigma \quad with\, x \neq y$
- Insertion if $x = - \; and \; y \in \Sigma$
- Deletion if $x \in \Sigma \; and \; y = -$

Edit Operation

Given two sequence $a, b \in (\Sigma \cup -)^*$

and *edit operation* (x,y) such that $x \neq y$

$$a \rightarrow_{(x,y)} b$$

means that b can be obtained from a by replacing on occurrence of x with y or by deleting on occurrence of x (if y = –)

If $S = s_1 \ldots s_r$ is a sequence of edit operations then

$$a \Rightarrow_S b$$

Edit Distance

Given a *cost function*

$$w : (\Sigma \cup \{-\}) \times (\Sigma \cup \{-\}) \to \Re$$

and two words $a, b \in \Sigma$, the cost of a sequence $S = s_1 \ldots s_r$ of edit operations is defined by

$$\sum_{i=1}^{r} w(s_i)$$

The *edit distance* of a,b is defined as

$$D(a,b) = d_w(a,b) = \min\{w(S) \mid a \Rightarrow_S b\}.$$

Metric

A *metric* d must satisfy

- d(x,y) = 0 iff x = y
- d(x,y) = d(y,x) (symmetry)
- d(x,z) $\leq$ d(x,y) + d(y,z) (triangle inequality)

A cost function must be a metric.

Alignment

An *alignment* is a pair of sequences (a°, b°) consisting of letters from the alphabet Σ or – such that

$$|a^\circ| = |b^\circ|$$

and there is no position I such that

$$a^\circ_i = - = b^\circ_i$$

Furthermore,

a° without $- = a$, and

b° without $- = b$

Alignment Distance

The *alignment distance* of a,b is

$$d_w^a(a,b) = D(a,b)$$

$$= \min\{w(a^\circ, b^\circ) \mid (a^\circ, b^\circ) \textit{ alignment of } (a,b)\}.$$

The alignment is optimal if

$$d_w^a(a,b) = w(a^\circ, b^\circ).$$

Properties of Aligments

- Additivity of alignments

$$w(a^\circ c^\circ, b^\circ d^\circ) = w(a^\circ, b^\circ) + w(c^\circ, d^\circ)$$

- For every alignment of (a,b) there is a sequence of edit operations S, such that

$$a \Rightarrow_S b \quad and$$

$$w(S) = w(a^\circ, b^\circ).$$

Example

Consider the alphabet Σ = {A, C, T, G} and the two words a=ACCGGTA and b=AGGCTG.

One possible alignment is

a° = ACCGG–TA

b° = A– –GGCTG

The sequence S of edit operations with the above properties is

S = (C, –)(C, –)(–,C)(A,G)

Pairwise Sequence Alignment

Typical operations:

- Find differences between two similar sequences
  - insertions, deletions, substitutions
  - may need to compare large sequences (over 10000 characters)
- Find local similarities
  - look for a few hundred characters in each string
  - need to identify partial matches
  - useful for searching large databases
- Is one sequence a prefix of another?
  - useful in DNA fragment assembly
- Find the similarities between two sequences with same evolutionary background

Gapped matching vs. ungapped matching

- Ungapped matching is less demanding than gapped matching. There is only one optimal way in which COMPARE and COMPLETE can be aligned without introducing gaps.
- Introducing gaps into either sequence means multiple permutations of the alignment are allowed
- Increase state or solution space

```
COMPARE      COMP-ARE     COMPARE
****         ****  *      ****  *
COMPLETE     COMPLETE     COMPL-ETE
```

Alignments

Given two sequences u and v, an alignment is a pair of sequences u' and v' such that:

1. u' is obtained from u by inserting gap character '-'
2. v' is obtained from v by inserting gap character '-'
3. u' and v' have same length: | u' | = | v' |
4. No position has gap characters in both u' and v'

Example:

u = ATGGCT
v = TGCTA
u' = ATGGCT-
v' = -TG-CTA

Goal: given two sequences, find the "best" alignment according some scoring function.

Dynamic Programming

- Compares two sequences and generates an alignment
- Alignment contains matched and mismatched characters as well as gaps
- Can be used for both local (Smith-Waterman) and global (Needleman-Wunch) alignments
- Generates an alignment score so that significance of or optimal alignment can be found
- Depends on choice of scoring system

Practical Considerations

- Goal of alignment will determine the type of scoring matrix used
- PAM based on model of evolutionary change
- BLOSUM are defined to identify members of the same family
- Different types of gap penalties

Concept of Distance or Similarity

- Distance
  - The distance between two sequences, based on an evolutionary model, describes when the two sequences had a common ancestor.
  - We want to minimize the distance.
- Similarity
  - The similarity between two sequence described how closely related two sequences are.
  - We want to maximize the similarity
- Either can be used and get similar results

Metrics

- Any notion of distance or similarity must be a metric
- A metric d must satisfy the following
  - D(x,y) = 0 if x = y
  - D(x,y) = D(y,x) (symmetry)
  - D(x,z) <= D(x,y) + D(y,z) (triangle inequality)
- The concept of distance between two points satisfies this (Euclidean distance or Euclidean metric)

How gapped matches are scored

- The scoring function is expanded to include a penalty for gaps
- The penalty value is generally chosen to be costly enough, in terms of the current scoring matrix, that adding a gap will not be too easy (resulting in meaningless alignments) or too difficult (resulting in no gaps).
- It costs less to extend a gap once it's opened than to open it in the first place.

```
ACGTAGTGT-CACT        -ACGTAGTGTCA-C-T
 *  ** **              * * * **  * * *
GAGA--TGAGCATG        GA-G-A-TG--AGCATG
```

Gap penalties

- Linear gap penalty function
- Subadditive gap penalty function
  $g(k+l) <= g(k) + g(l)$
- Affine gap penalty function
  $g(k) = a + kb$
- Different gap penalty at the ends of sequences

Steps to Dynamic Programming

- Compute the similarity/distance matrix for two sequences
- Perform the trace back to find the optimal alignment
- We can use distance or similarity and get the same result

Example: Consider the words a = AT and b = AAGT and a similarity score function s(x,y) = 0 if x >< y and w(x,x) =1

Example: Consider the words a = AT and b = AAGT and a cost score function d(x,y) = 1 if x >< y and w(x,x) = 0

- In these examples, the gap and mismatch penalty are equal.
- The text minimizes similarity in their algorithm

Needleman-Wunch Algorithm (Global Alignment)

$$D_{0,,0} = 0$$

$$D_{0,j} = \sum_{k=1}^{j} w(-, b_k)$$

$$D_{i,0} = \sum_{k=1}^{i} w(a_k, -)$$

$$\forall i, j > 0 \quad D_{i,j} = \min \begin{Bmatrix} D_{i,j-1} + w(-, b_j) \\ D_{i-1,j-1} + w(a_i, b_j) \\ D_{i-1,j} + w(a_i, -) \end{Bmatrix}$$

where w is the weight of a gap

Needleman-Wunch Algorithm Example

Example: Consider the words a = AT and b = AAGT and a cost function s(x,y) = 1 if x ><y and w(x,x) =0

Needleman-Wunch Algorithm Example

		A	A	G	T
A					
T					

Needleman-Wunch Algorithm Example

		A	A	G	T
	0	1	2	3	4
A	1				
T	2				

Needleman-Wunch Algorithm Example

		A	A	G	T
	0	1	2	3	4
A	1	0 2			
		2 0			
T	2				

Needleman-Wunch Algorithm Example

		A	A	G	T
	0	1	2	3	4
A	1	0 2			
		2 0			
T	2	2 1			
		3 1			

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3				
		2	0	1	1				
T	2	2	1						
		3	1						

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3				
		2	0	1	1				
T	2	2	1	1	2				
		3	1	2	1				

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4		
		2	0	1	1	2	2		
T	2	2	1	1	2				
		3	1	2	1				

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4		
		2	0	1	1	2	2		
T	2	2	1	1	2	2	3		
		3	1	2	1	2	2		

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4	4	5
		2	0	1	1	2	2	3	3
T	2	2	1	1	2	2	3		
		3	1	2	1	2	2		

Needleman-Wunch Algorithm Example

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4	4	5
		2	0	1	1	2	2	3	3
T	2	2	1	1	2	2	3	2	4
		3	1	2	1	2	2	3	2

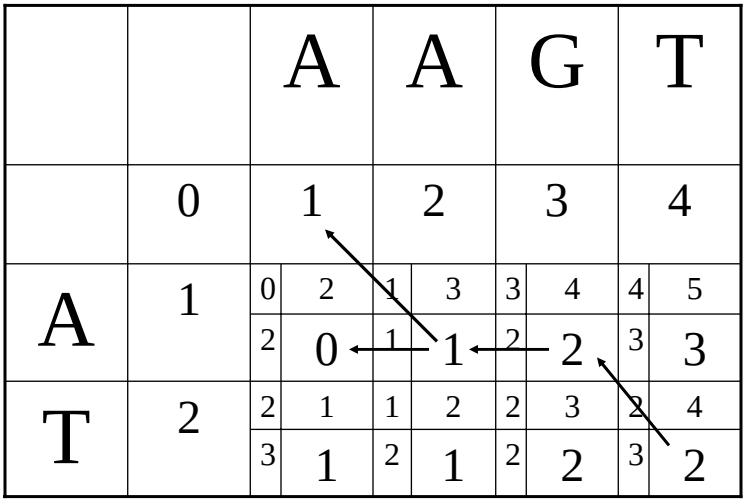
Traceback

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4	4	5
		2	0	1	1	2	2	3	3
T	2	2	1	1	2	2	3	2	4
		3	1	2	1	2	2	3	2

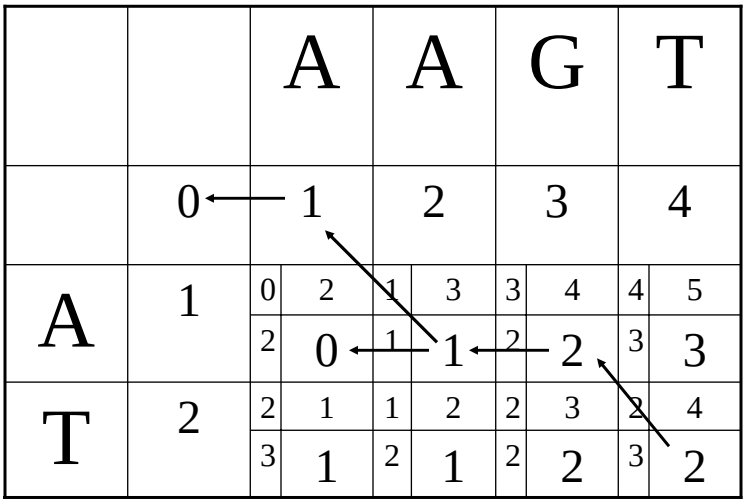
Traceback

		A		A		G		T	
	0	1		2		3		4	
A	1	0	2	1	3	3	4	4	5
		2	0	1	1	2	2	3	3
T	2	2	1	1	2	2	3	2	4
		3	1	2	1	2	2	3	2

Traceback



Traceback



Traceback

		A	A	G	T
	0	1	2	3	4
A	1	0 2 1 3 3 4 4 5 2 0 1 1 2 2 3 3			
T	2	2 1 1 2 2 3 3 4 3 1 2 1 2 2 3 2			

Getting Alignment from Trace back

- The alignments can be determined from the traceback
- Vertical arrows denote a gap in the sequence on the top
- Horizontal arrows denote a gap in the sequence on the right
- Diagonal arrows denote a match if there is not penalty
- Diagonal arrows denote a mismatch if there is a penalty

Alignment

AAGT
– A –T

		A	A	G	T
	0 ←	1	2	3	4
A	1	0 2 1 3 3 4 4 5 2 0 ← 1 ← 2 3 3			
T	2	2 1 1 2 2 3 2 4 3 1 2 1 2 2 3 2			

Alignment

AAGT
A– – T

		A	A	G	T
	0	1	2	3	4
A	1	0 2 1 3 3 4 4 5 2 0 ← 1 ← 2 3 3			
T	2	2 1 1 2 2 3 2 4 3 1 2 1 2 2 3 2			

Waterman-Smith-Beyer Algorithm (Local Alignment)

$$D_{0,,0} = 0$$

$$D_{0,j} = g(j)$$

$$D_{i,0} = g(i)$$

$$\forall i, j > 0 \quad S_{i,j} = \min\begin{Bmatrix} \min_{1 \le k \le j}(D_{i,j-k} + g(k)) \\ D_{i-1,j-1} + w(a_i, b_j) \\ \min_{1 \le k \le j}(D_{i-k,j} + g(k)) \end{Bmatrix}$$

where $g(k)$ is the gap penalty function and
w is the similarity score function

Waterman-Smith-Beyer Algorithm Example

Example: Consider the words a = AT and b = AAGT and a cost function s(x,y) = 1 if substitution and s(x,x) =0. We will assume an affine gap penalty function g(k) = 2 + k

Waterman-Smith-Beyer Algorithm Example

		A		A		G		T	
	0	2		3		4		5	
A	2	0	4	2	5	4	6	5	7
		4	0	2	2	4 3	3	4	4
T	3	3	2	1	4	3	5	3	6
		5	2	4	1	3	3	4 5	3

Waterman-Smith-Beyer Algorithm Example

		A		A		G		T	
	0	2		3		4		5	
A	2	0	4	2	5	4	6	5	7
		4	0	2	2	4 3	3	4	4
T	3	3	2	1	4	3	5	3	6
		5	2	4	1	3	3	4 5	3

Alignment

AAGT
A– – T

		A	A	G	T
	0	2	3	4	5
A	2	0 4	2 5	4 6	5 7
	4	0	2	4 3 3	4 4
T	3	3 2	1 4	3 5	3 6
	5	2	4 1	3 3	4 5 3

Alignment (Not optimal)

AAGT
– A –T

		A	A	G	T
	0	2	3	4	5
A	2	0 4	2 5	4 6	5 7
	4	0	2	4 3 3	4 4
T	3	3 2	1 4	3 5	3 6
	5	2	4 1	3 3	4 5 3

Enhancements to Dynamic Programming

- Needleman and Wunch (1970) – global alignment
- Smith and Waterman (1980) – local alignment ie. alignment does not have to start at the ends
- Gotoh (1982) – decreased number of steps
- Waterman and Eggert (1987) – find alternative alignments ie., can start alignment in different places
- Myers and Miller (1988) – decreased memory required
- Schwartz (1991)– long sequence alignment
- Chao (1994) – near-optimal alignments

These methods are constantly evolving.

Significance of Alignment

- Dayhoff evaluated Needleman-Wunch alignment scores for many randomized and unrelated protein sequences using their log odds scoring matrix at 250 PAMs and a constant gap penalty.
- Result were normally distributed.
- For a score of an alignment to be significant, it must be at least 3-5 standard deviations greater than the mean of the random scores
- Caveats: computationally expensive and assumes random distribution of characters in alphabet.