"You better hurry. Management wants the data cleaned up by tomorrow morning."

Lecture 5: Aligning Sequences

The issue: which are the corresponding sites in sequences thought to be related (homologous)?

Reference: Higgins & Lemey (2009) Chapter 3 of *The Phylogenetic Handbook*. Cambridge University Press.

Alignment Example:

Two short sequence segments of the 28S rRNA gene from the human and the carp.

Site	`12345678901234567890`
Human	`CGGCAAGGCTTCCCTGCCGG`
Carp	`CGGTCAAGCCTTCCCTCCGG`

Human and Carp Alignment #1

Substitutions only

```
Site   12345678901234567890
Human  CGGCAAGGCTTCCCTGCCGG
Carp   CGGTCAAGCCTTCCCTCCGG
```

The alignment implied by this way of arranging these two sequences requires that at least 7 substitutions occurred in their common history.

Human and Carp Alignment #2

Substitutions and Gaps Allowed

```
Site    123456789012345678901
Human   CGG*CAAGGCTTCCCTGCCGG
Carp    CGGTCAAGCCTTCCCT*CCGG
```

This alignment requires two gaps (insertions or deletions denoted by a "*") but only one substitution.

If insertions or deletions are common, then the second alignment should be preferred, but if they are rare, then gaps should be heavily penalized and the first alignment may be preferred.

Aligning Sequences

The choice of alignment then depends on minimizing a score function that specifies appropriate weights for different types of substitutions and gaps.

A typical score function gives 0 points per match, 1 point for needing a substitution, b_1 points for opening a gap and b_2 points for each position a gap is extended.

The optimal alignment can then be found using the dynamic programming technique.

Aligning Sequences

The Dot Matrix Technique

The “Dot matrix” method can then used to visually spot good alignments.

One sequence forms the columns and the other forms the rows of a matrix. Dots are put in the matrix where the nucleotides are identical.

Longer sequences can be divided into blocks with similarity scores coded in the matrix.

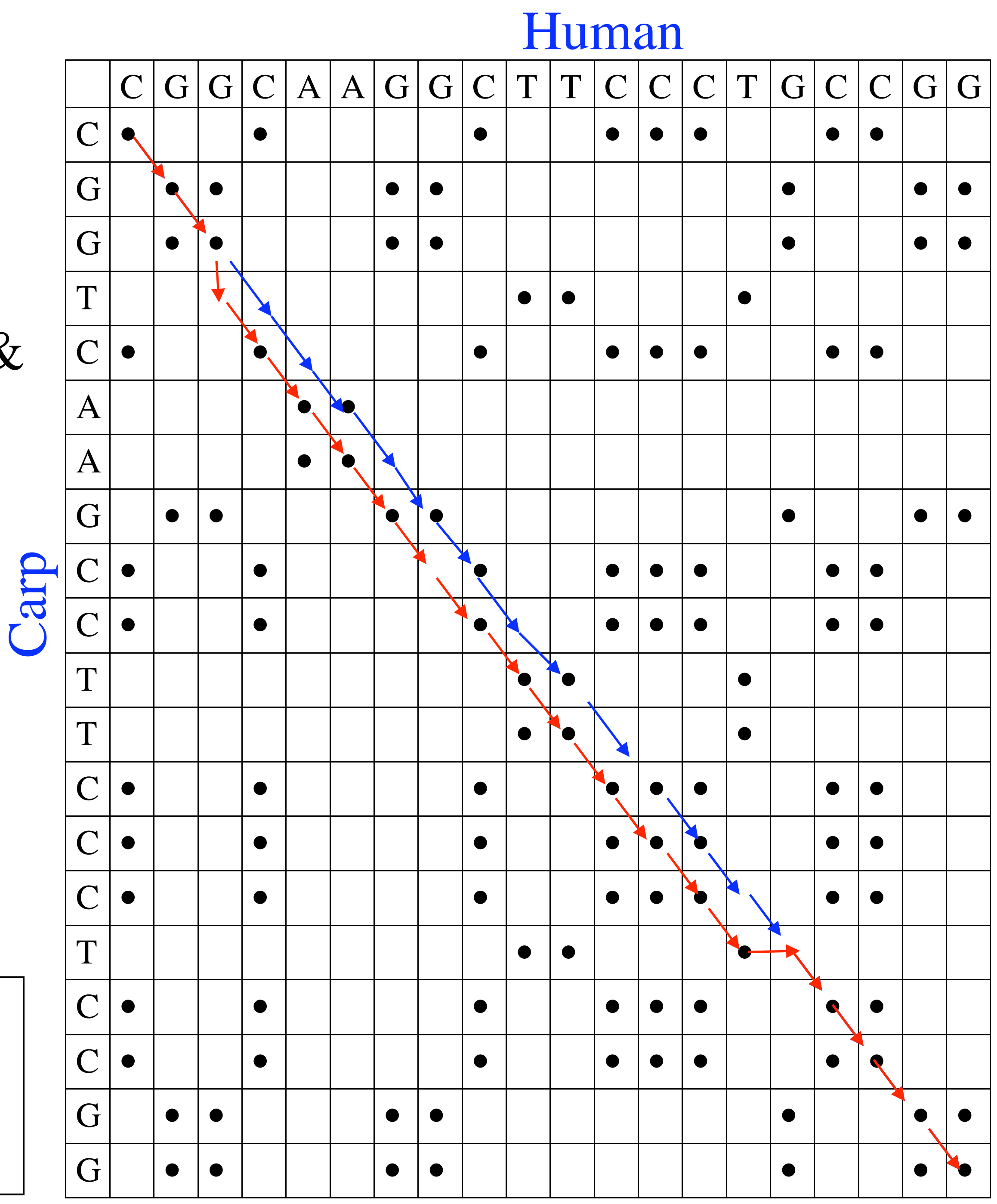
Dot Matrix aligning strings of the human & Carp 28S rRNA gene

An alignment = a path from the upper left to the lower right.

Blue path = alignment 1

Red path = alignment 2

↘	No gap
→	Gap in row sequence
↓	Gap in column sequence



	C	G	G	C	A	A	G	G	C	T	T	C	C	C	T	G	C	C	G	G
C	●			●					●			●	●	●			●	●		
G		●	●				●	●								●			●	●
G		●	●				●	●								●			●	●
T										●	●				●					
C	●			●					●			●	●	●			●	●		
A					●	●														
A					●	●														
G		●	●				●	●								●			●	●
C	●			●					●			●	●	●			●	●		
C	●			●					●			●	●	●			●	●		
T										●	●				●					
T										●	●				●					
C	●			●					●			●	●	●			●	●		
C	●			●					●			●	●	●			●	●		
C	●			●					●			●	●	●			●	●		
T										●	●				●					
C	●			●					●			●	●	●			●	●		
C	●			●					●			●	●	●			●	●		
G		●	●				●	●								●			●	●
G		●	●				●	●								●			●	●

Notation

Want to align sequence **a** having nucleotide

a_i at position i for i = 1,..., N

with sequence **b** having nucleotide

b_j at position j for j = 1,...., M.

An alignment consists of sequences **a*** and **b***, each of length K, whose elements are either gaps or the original elements of **a** and **b**.

In the human versus carp 28S rRNA example N=M=20,

a = `CGGCAAGGCTTCCCTGCCGG`

b = `CGGTCAAGCCTTCCCTCCGG`

for the second alignment K = 21,

```
a* =  CGG-CAAGGCTTCCCTGCCGG
b* =  CGGTCAAGCCTTCCCT-CCGG
```

The optimality problem

- Note can't align blank with blank so that

$$N+M \geq K \geq \max\{N,M\}.$$

- The problem is to produce **a*** and **b*** in an optimal way…. for example using the criterion

$$\sum_{k=1}^{K} \delta(a_k^*, b_k^*)$$

- The score function δ may be a measure of similarity (the maximum is optimal) or of distance (the minimum is optimal) between the sequences **a*** and **b***.

For example, we could use the log-likelihood from a stochastic model that assumes site-to-site independence

Dynamic Programming for Alignment

Bellman, R. (1957). Dynamic Programming. Princeton University Press.

Needleman, S.B., and Wunsch, C.D. (1970). A general method applicable to the search for similarities in the amino acid sequences of two proteins. *Journal of Molecular Biology* **48:** 443-453.

- Define S(n,m) to be the optimal score aligning sequence **a** up to position n with sequence **b** up to position m for $0 \leq n \leq N$, $0 \leq m \leq M$.

- A global alignment of the two sequences **a** and **b** finds **a*** and **b*** which have a total score of S(N,M).

Dynamic Programming for Alignment

- To find S(N,M) you fill in the matrix of S(n,m) values recursively using

$$S(n,m) = \max \begin{cases} S(n-1,m-1) + \delta(a_n,b_n) \\ S(n-1,m) + \delta(a_n,gap) \\ S(n,m-1) + \delta(gap,b_n) \end{cases}$$

Need to pick a δ function and initialize S.

Example (α<0):

$$\delta(x,y) = \begin{cases} 1 & \text{if } x = y \neq \text{gap} \\ \alpha & \text{if x or y = gap} \\ 0 & \text{otherwise} \end{cases}$$

S(0,0) = 0   S(n,0) = $n\alpha$ S(0,m) = $m\alpha$

Dynamic Programming Matrix for aligning strings of the human & Carp 28S rRNA gene (simple δ and $\alpha = -2$)

Human

Carp

		C	G	G	C	A	A	G	G	C	T	T	C	C	C	T	G	C	C	G	G
	0	-2	-4	-6	**-8**	-10	-12	-14	-16	**-18**	-20	-22	**-24**	**-26**	**-28**	-30	-32	**-34**	**-36**	-38	-40
C	**-2**	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19	-21	-23	-25	-27	-29	-31	-33	-35	-37
G	-4	-1	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24	-26	-28	-30	-32	-34
G	-6	-3	0	3	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19	-21	-23	-25	-27	-29	-31
T	-8	-5	-2	1	3	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19	-21	-23	-25	-27	-29
C	**-10**	-7	-4	-1	2	3	1	-1	--3	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24	-26
A	-12	-9	-6	-3	0	3	4	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24
A	-14	-11	-8	-5	-2	1	4	4	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22
G	-16	-13	-10	-7	-4	-1	2	5	5	3	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19
C	**-18**	-15	-12	-9	-6	-3	0	3	5	6	4	2	0	-2	-4	-6	-8	-10	-12	-14	-26
C	**-20**	-17	-14	-11	-8	-5	-2	1	3	6	6	4	3	1	-1	-3	-5	-7	-9	-11	-13
T	-22	-19	-16	-13	-10	-7	-4	-1	1	4	7	7	5	3	1	0	-2	-4	-6	-8	-10
T	-24	-21	-18	-15	-12	-9	-6	-3	-1	2	5	8	7	5	3	2	0	-2	-4	-6	-8
C	**-26**	-23	-20	-17	-14	-11	-8	-5	-3	0	3	6	9	8	6	4	2	1	-1	-3	-5
C	**-28**	-25	-22	-19	-16	-13	-10	-7	-5	-2	1	4	7	10	9	7	5	3	2	0	-2
C	**-30**	-27	-24	-21	-18	-15	-12	-9	-7	-4	-1	2	5	8	11	9	7	6	4	2	0
T	-32	-29	-26	-23	-20	-17	-14	-11	-9	-6	-3	0	3	6	9	12	10	8	6	4	2
C	**-34**	-31	-28	-25	-22	-19	-16	-13	-11	-8	-5	-2	1	4	7	9	12	11	9	7	5
C	**-36**	-33	-30	-27	-24	-21	-18	-15	-13	-10	-7	-4	-1	2	5	7	10	13	12	10	8
G	-38	-35	-32	-29	-26	-23	-20	-17	-15	-12	-9	-6	-3	0	3	5	8	11	13	13	11
G	-40	-37	-34	-31	-28	-25	-22	-19	-17	-14	-11	-8	-5	-2	1	3	6	9	11	14	14

Dynamic Programming algorithm

- S(N,M) will be the optimal score.
- To find an alignment that gives the optimal score (note – there will be at least one) you need to save pointers from each cell in the matrix back to the cell from which it was computed.
- Any path from (N,M) back to (0,0) that follows these pointers will be an optimal alignment. This is called the *traceback* procedure.

Dynamic Programming Matrix for aligning strings of the human & Carp 28S rRNA gene (simple δ and $\alpha = -2$)

Human

Carp

		C	G	G	C	A	A	G	G	C	T	T	C	C	C	T	G	C	C	G	G
	0	-2	-4	-6	**-8**	-10	-12	-14	-16	**-18**	-20	-22	**-24**	**-26**	**-28**	-30	-32	**-34**	**-36**	-38	-40
C	**-2**	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19	-21	-23	-25	-27	-29	-31	-33	-35	-37
G	-4	-1	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24	-26	-28	-30	-32	-34
G	-6	-3	0	3	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19	-21	-23	-25	-27	-29	-31
T	-8	-5	-2	1	3	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19	-21	-23	-25	-27	-29
C	**-10**	-7	-4	-1	2	3	1	-1	--3	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24	-26
A	-12	-9	-6	-3	0	3	4	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24
A	-14	-11	-8	-5	-2	1	4	4	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22
G	-16	-13	-10	-7	-4	-1	2	5	5	3	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19
C	**-18**	-15	-12	-9	-6	-3	0	3	5	6	4	2	0	-2	-4	-6	-8	-10	-12	-14	-26
C	**-20**	-17	-14	-11	-8	-5	-2	1	3	6	6	4	3	1	-1	-3	-5	-7	-9	-11	-13
T	-22	-19	-16	-13	-10	-7	-4	-1	1	4	7	7	5	3	1	0	-2	-4	-6	-8	-10
T	-24	-21	-18	-15	-12	-9	-6	-3	-1	2	5	8	7	5	3	2	0	-2	-4	-6	-8
C	**-26**	-23	-20	-17	-14	-11	-8	-5	-3	0	3	6	9	8	6	4	2	1	-1	-3	-5
C	**-28**	-25	-22	-19	-16	-13	-10	-7	-5	-2	1	4	7	10	9	7	5	3	2	0	-2
C	**-30**	-27	-24	-21	-18	-15	-12	-9	-7	-4	-1	2	5	8	11	9	7	6	4	2	0
T	-32	-29	-26	-23	-20	-17	-14	-11	-9	-6	-3	0	3	6	9	12	10	8	6	4	2
C	**-34**	-31	-28	-25	-22	-19	-16	-13	-11	-8	-5	-2	1	4	7	9	12	11	9	7	5
C	**-36**	-33	-30	-27	-24	-21	-18	-15	-13	-10	-7	-4	-1	2	5	7	10	13	12	10	8
G	-38	-35	-32	-29	-26	-23	-20	-17	-15	-12	-9	-6	-3	0	3	5	8	11	13	13	11
G	-40	-37	-34	-31	-28	-25	-22	-19	-17	-14	-11	-8	-5	-2	1	3	6	9	11	14	14

Dynamic Programming algorithm: Computational complexity

- The algorithm requires that we store (N+1)(M+1) numbers in the matrix plus less than 3(N+1)(M+1) pointers and each number in the matrix requires a constant number of calculations to compute (looking at 3 sums to find a maximum). Thus the algorithm takes O(NM) time and O(MN) memory.
- The algorithm is guaranteed to give an optimal alignment (not necessarily unique) and the optimal score.

Aligning Multiple Sequences

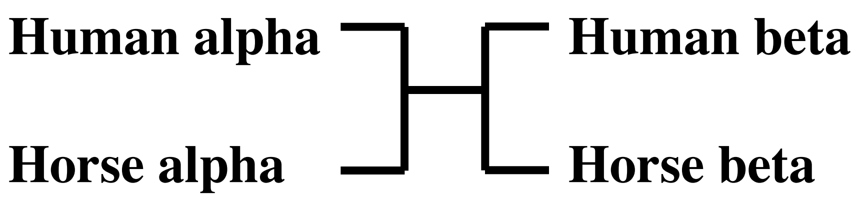
When there are many sequences, it is important to also account for their phylogenetic history.

Site	`123456789012345678`
Human alpha	`AAWGKVGAHAGEYGAEAL`
Horse alpha	`ALWSKVGGHAGEYGAEAL`
Human beta	`ALWGKVNVDEVGGEALAV`
Horse beta	`ALWDKVNEEEVGGEALAV`

Aligning Multiple Sequences

When there are many sequences, it is important to also account for their phylogenetic history.

Site	12345678901234567890
Human alpha	AAWGKVGAHAGEYGAEAL**
Horse alpha	ALWSKVGGHAGEYGAEAL**
Human beta	ALWGKVN**VDEVGGEALAV
Horse beta	ALWDKVN**EEEVGGEALAV



Aligning Multiple Sequences

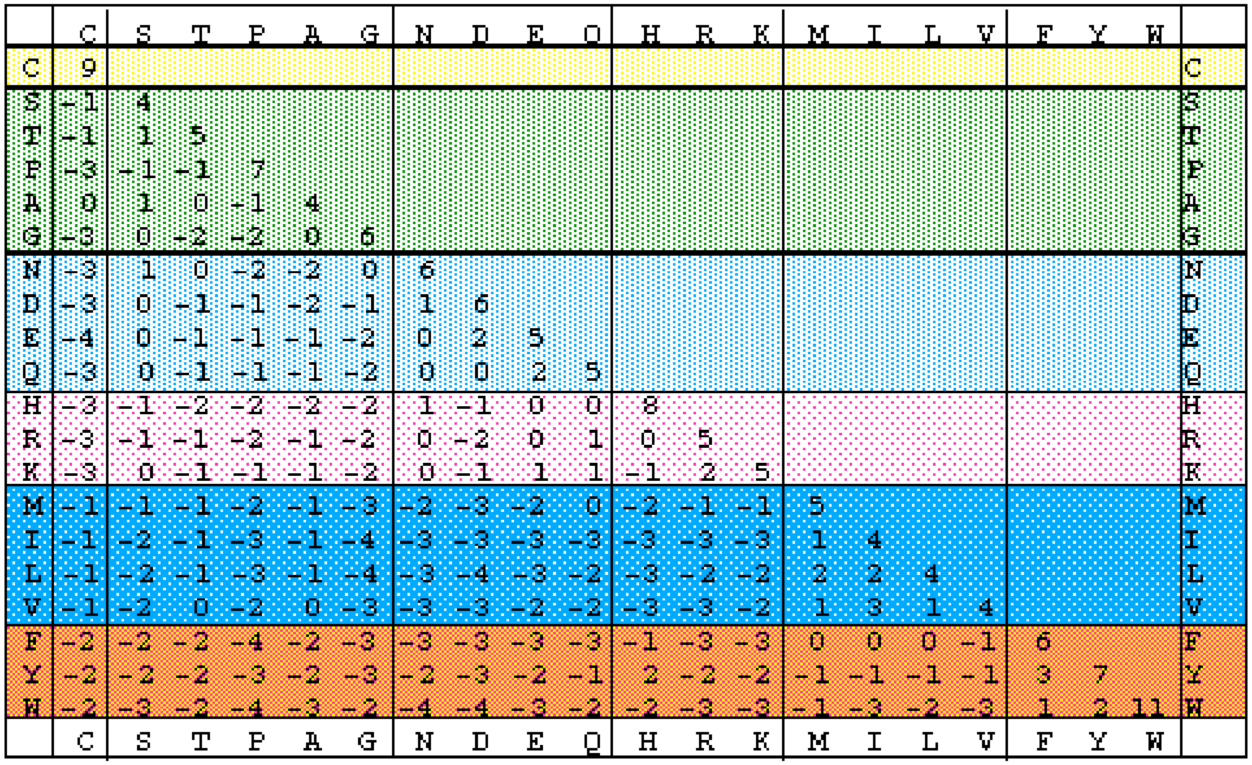
- In coding sequences alignments are generally based on the amino acid sequences.
- Scoring Example: $S(m) = G + \sum_i S(m_i)$
- Gap functions are typically affine functions that penalize gap initiation more than gap continuation.

	Nucleotides	Amino acids
Match	0	0
Mismatch	-1	BLOSUM62
Gap Start	-10	-11
Gap Continuation	-0.1	-1

- Column scores might measure entropy
  - E.g. $S(m_i) = \sum_j c_{ij} \log(\hat{p}_{ij})$ (sum is over acids j in column i)

Empirical Weight Matrix Example

BLOSUM62



	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	9																				C
S	-1	4																			S
T	-1	1	5																		T
P	-3	-1	-1	7																	P
A	0	1	0	-1	4																A
G	-3	0	-2	-2	0	6															G
N	-3	1	0	-2	-2	0	6														N
D	-3	0	-1	-1	-2	-1	1	6													D
E	-4	0	-1	-1	-1	-2	0	2	5												E
Q	-3	0	-1	-1	-1	-2	0	0	2	5											Q
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8										H
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5									R
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5								K
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5							M
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4						I
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4					L
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4				V
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6			F
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	7		Y
W	-2	-3	-2	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	11	W
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	

Dynamic Programming algorithm:

- Define $S(m_1, m_2, \ldots, m_n)$ to be the optimal score aligning sequence $\mathbf{a_1}$ up to position m_1, with sequence $\mathbf{a_2}$ up to position m_2, ..., and with sequence $\mathbf{a_n}$ up to position m_n.
- A global alignment of the n sequences $\mathbf{a_1}$ to $\mathbf{a_n}$ finds $\mathbf{a_1}$* to $\mathbf{a_n}$* which have a total score of $S(K_1, \ldots, K_n)$.

Dynamic Programming algorithm:

- To find $S(K_1,\ldots,K_n)$ you fill in the n dimensional array recursively by $S(m_1, m_2, \ldots,m_n) =$

$$\max \left[\begin{array}{l} S(m_1 - 1,\ldots,m_n - 1) \ + \ \delta(a_{m_1},\ldots,a_{m_n}) \\ S(m_1, m_2 - 1,\ldots,m_n - 1) \ + \ \delta(gap, a_{m_2},\ldots,a_{m_n}) \\ \qquad\qquad\qquad\qquad \ldots \\ S(m_1 - 1, m_2 - 1,\ldots,m_n) \ + \ \delta(a_{m_1},\ldots,a_{m_{n-1}}, gap) \\ S(m_1, m_2,\ldots,m_n - 1) \ + \ \delta(gap, gap, a_{m_3},\ldots,a_{m_n}) \\ \qquad\qquad\qquad\qquad \ldots \\ S(m_1 - 1, m_2,\ldots,m_n) \ + \ \delta(a_{m_1}, gap,\ldots,gap) \end{array} \right.$$

Dynamic Programming algorithm:

- To find an alignment with the optimal score you need to save pointers from each cell in the array back to the cell from which it was computed.
- Traceback: Any path from $(K_1,\ldots,K_n)$ back to $(0,\ldots,0)$ that follows these pointers will be an optimal alignment.
- If scoring system is monotone in specific directions than bounds can be used to limit the complexity of the algorithm (analogous to branch-and-bound method for phylogeny).

Dynamic Programming algorithm: Computational complexity

- The algorithm requires storage of $\prod_{i=1}^{n}(K_i+1)$ numbers in the array. To calculate each entry we must maximize over all 2^n-1 combinations of gaps in a column. Thus the algorithm takes $O(\overline{K}^n)$ memory and $O(2^n\overline{K}^n)$ time.

- The algorithm is guaranteed to give an optimal alignment (not necessarily unique) and the optimal score.

Basic “Progressive” algorithm:

- Step 1: Use dynamic programming to find all pairwise alignments. Use these alignments to create a distance matrix (e.g. Kimura distances)
- Step 2: Use the distance matrix to construct a guide tree using a fast algorithm like the neighbor-joining tree method.
- Step 3: progressively align at interior nodes in order of decreasing similarity. Interior node alignments require a sequence-to-sequence, sequence-to-alignment or alignment-to-alignment algorithms.

Issues:

- Need better agreement with full tree
- Improvements can be gained by knowing protein structure
- Alignment in the framework of a full stochastic model
- Should account for a distribution of alignments rather than a fixed single alignment
- Need ways to speed algorithms computationally