# Multiple Sequence Alignment

# Multiple Alignment versus Pairwise Alignment

- **Up until now we have only tried to align two sequences.**
- **What about more than two? And what for?**
- **A faint similarity between two sequences becomes significant if present in many**
- **Multiple alignments can reveal subtle similarities that pairwise alignments do not reveal**

# Generalizing the Notion of Pairwise Alignment

- Alignment of 2 sequences is represented as a 2-row matrix
- In a similar way, we represent alignment of 3 sequences as a 3-row matrix

```
A T _ G C G _
A _ C G T _ A
A T C A C _ A
```

- Score: more conserved columns, better alignment

# Alignments = Paths in…

- Align 3 sequences: ATGC, AATC,ATGC

| | | | | | |
|---|---|---|---|---|---|
| | A | -- | T | G | C |

| | | | | | |
|---|---|---|---|---|---|
| | A | A | T | -- | C |

| | | | | | |
|---|---|---|---|---|---|
| | -- | A | T | G | C |

# Alignment Paths

| 0 | 1 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| | A | -- | T | G | C |

| | | | | | |
|---|---|---|---|---|---|
| | A | A | T | -- | C |

| | | | | | |
|---|---|---|---|---|---|
| | -- | A | T | G | C |

*x* coordinate

# Alignment Paths

- Align the following 3 sequences:

ATGC, AATC,ATGC

| | | | | | |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 2 | 3 | 4 |
| | A | -- | T | G | C |
| 0 | 1 | 2 | 3 | 3 | 4 |
| | A | A | T | -- | C |
| | -- | A | T | G | C |

*x* coordinate

*y* coordinate

-

# Alignment Paths

| | | | | | |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 2 | 3 | 4 |
| | A | -- | T | G | C |
| 0 | 1 | 2 | 3 | 3 | 4 |
| | A | A | T | -- | C |
| 0 | 0 | 1 | 2 | 3 | 4 |
| | -- | A | T | G | C |

*x* coordinate

*y* coordinate

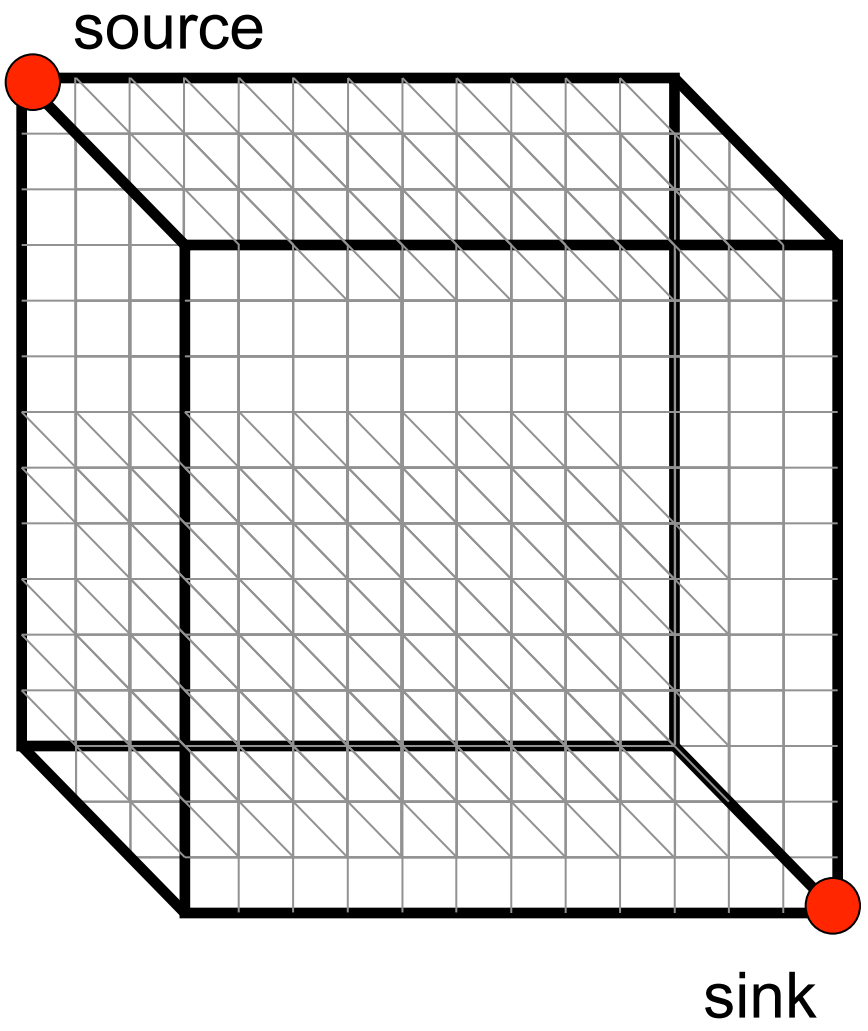
*z* coordinate

- Resulting path in *(x,y,z)* space:

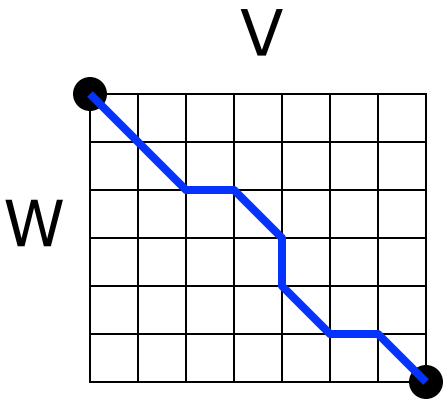
(0,0,0)→(1,1,0)→(1,2,1) →(2,3,2) →(3,3,3) →(4,4,4)

# Aligning Three Sequences

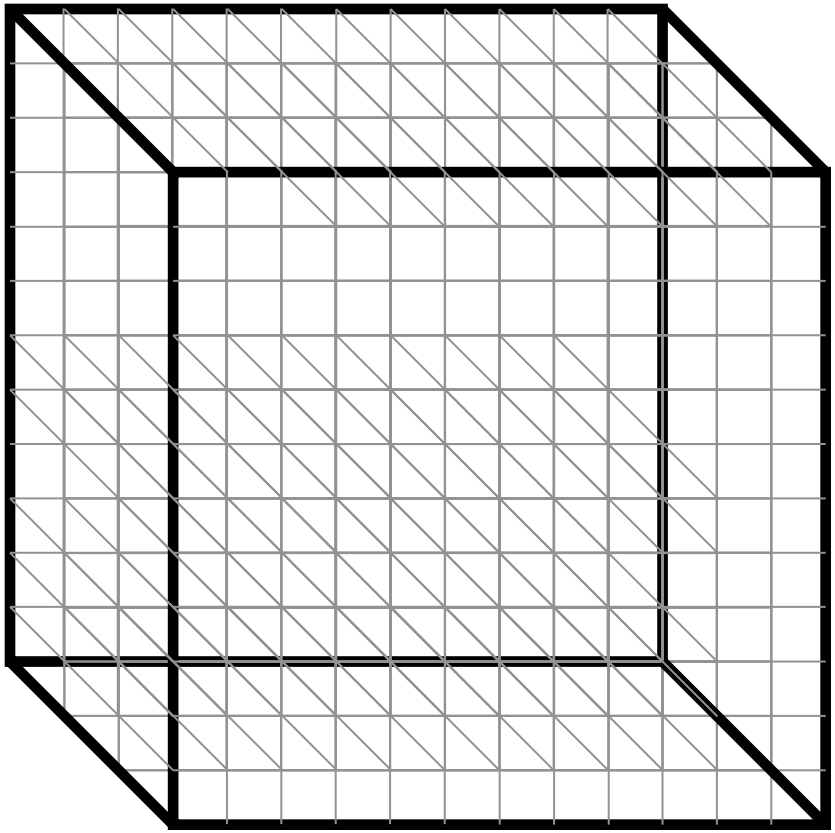
- Same strategy as aligning two sequences
- Use a 3-D “Manhattan Cube”, with each axis representing a sequence to align
- For global alignments, go from source to sink



# 2-D vs 3-D Alignment Grid

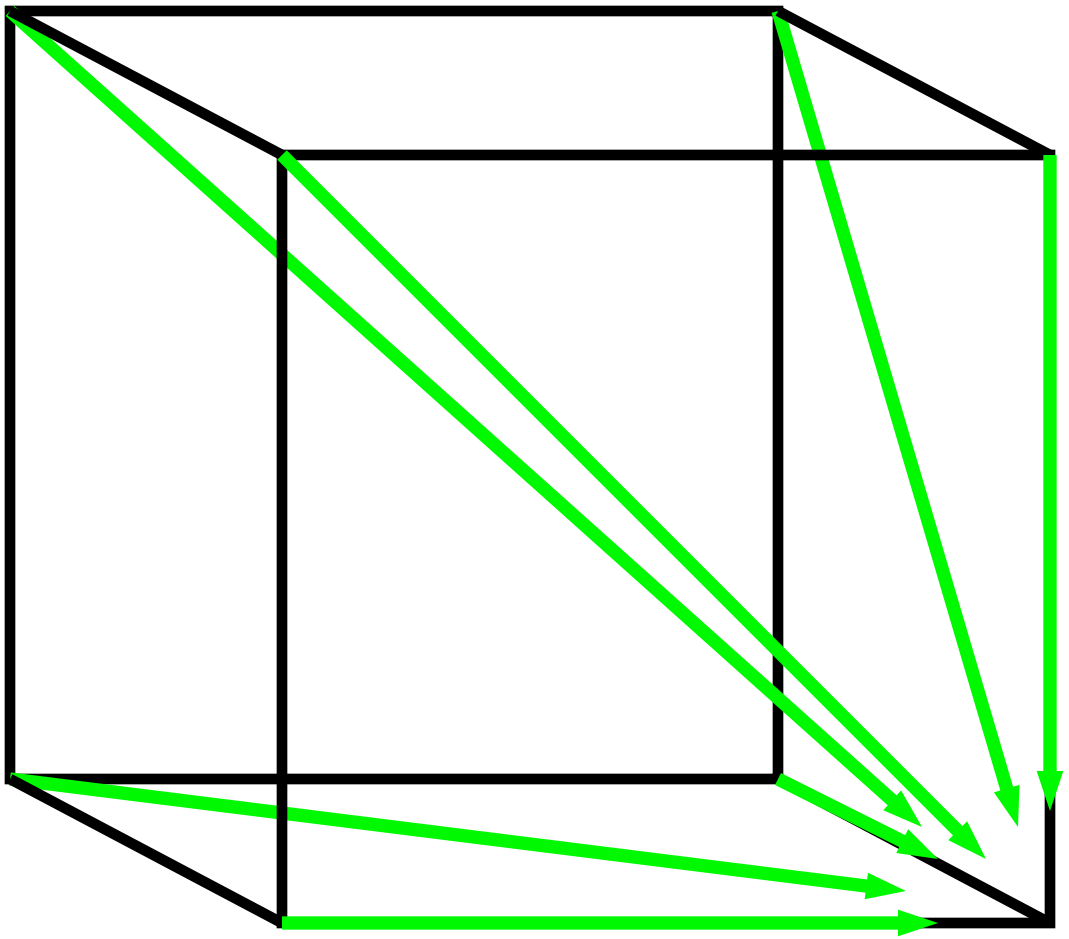


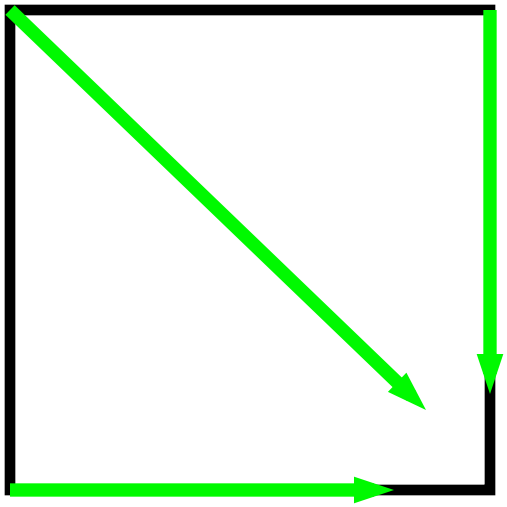

2-D edit graph



3-D edit graph

# 2-D cell versus 2-D Alignment Cell

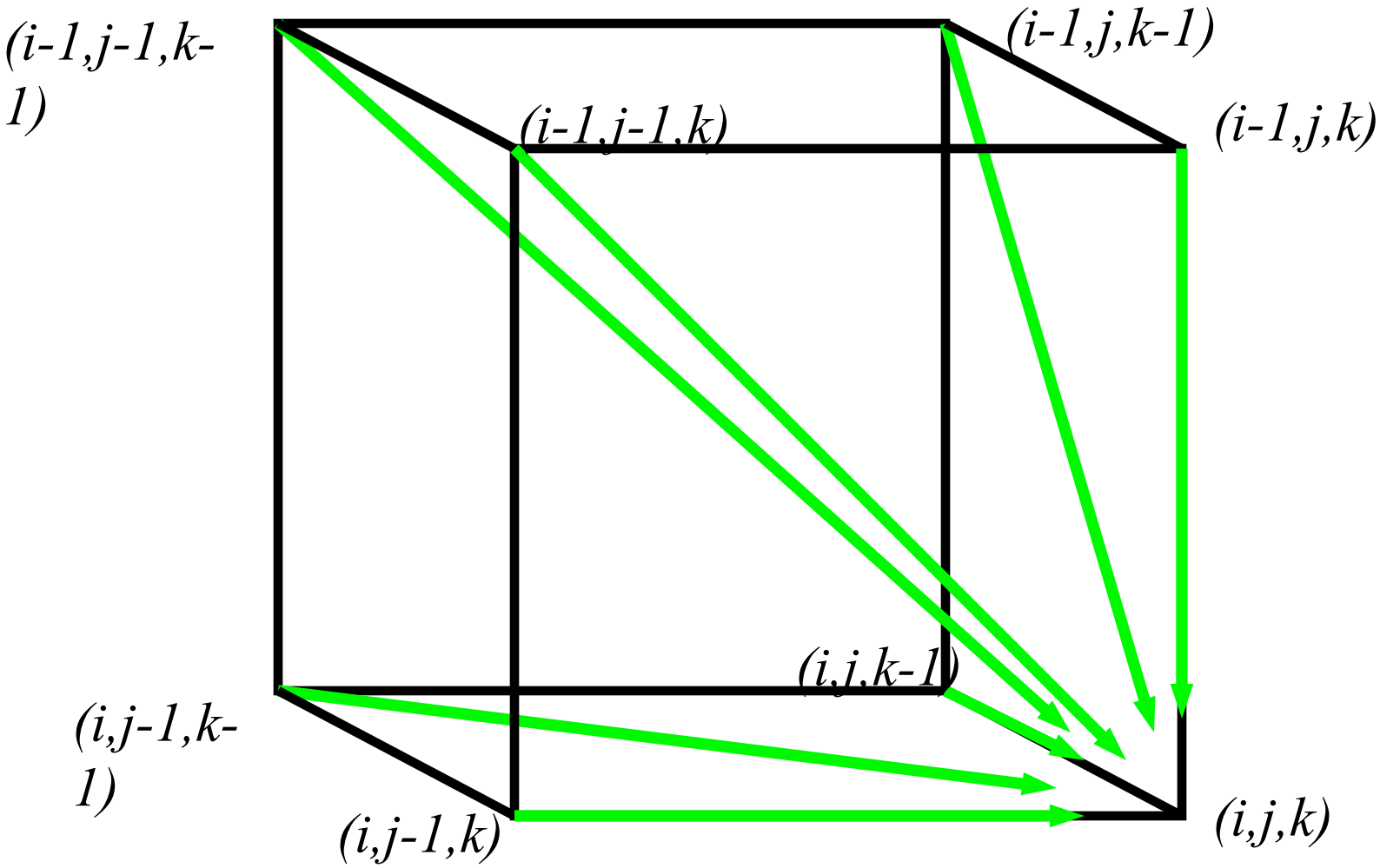




In 2-D, 3 edges in each unit square

In 3-D, 7 edges in each unit cube

# Architecture of 3-D Alignment Cell



# Multiple Alignment: Dynamic Programming

- $$s_{i,j,k} = \max \begin{cases} s_{i-1,j-1,k-1} + \delta(v_i, w_j, u_k) & \text{cube diagonal: no indels} \\ s_{i-1,j-1,k} + \delta(v_i, w_j, \_) & \\ s_{i-1,j,k-1} + \delta(v_i, \_, u_k) & \text{face diagonal: one indel} \\ s_{i,j-1,k-1} + \delta(\_, w_j, u_k) & \\ s_{i-1,j,k} + \delta(v_i, \_, \_) & \\ s_{i,j-1,k} + \delta(\_, w_j, \_) & \text{edge diagonal: two indels} \\ s_{i,j,k-1} + \delta(\_, \_, u_k) & \end{cases}$$

- $\delta(x, y, z)$ is an entry in the 3-D scoring matrix

# Multiple Alignment: Running Time

- For 3 sequences of length $\boldsymbol{n}$, the run time is $7\boldsymbol{n}^3$; $O(\boldsymbol{n}^3)$
- For $\boldsymbol{k}$ sequences, build a $\boldsymbol{k}$-dimensional Manhattan, with run time $(2^{\boldsymbol{k}}-1)(\boldsymbol{n}^{\boldsymbol{k}})$; $O(2^{\boldsymbol{k}}\boldsymbol{n}^{\boldsymbol{k}})$
- Conclusion: dynamic programming approach for alignment between two sequences is easily extended to $\boldsymbol{k}$ sequences but it is impractical due to exponential running time

# Multiple Alignment Induces Pairwise Alignments

Every multiple alignment induces pairwise alignments

```
x:  AC-GCGG-C
y:  AC-GC-GAG
z:  GCCGC-GAG
```

Induces:

```
x: ACGCGG-C;   x: AC-GCGG-C;   y: AC-GCGAG
y: ACGC-GAG;   z: GCCGC-GAG;   z: GCCGCGAG
```

# Reverse Problem: Constructing Multiple Alignment from Pairwise Alignments

**Given 3 arbitrary pairwise alignments:**

```
x: ACGCTGG-C;   x: AC-GCTGG-C;   y: AC-GC-GAG
y: ACGC--GAG;   z: GCCGCA-GAG;   z: GCCGCAGAG
```

**can we construct a multiple alignment that induces them?**

# Reverse Problem: Constructing Multiple Alignment from Pairwise Alignments

**Given 3 arbitrary pairwise alignments:**

```
x: ACGCTGG-C;   x: AC-GCTGG-C;   y: AC-GC-GAG
y: ACGC--GAG;   z: GCCGCA-GAG;   z: GCCGCAGAG
```

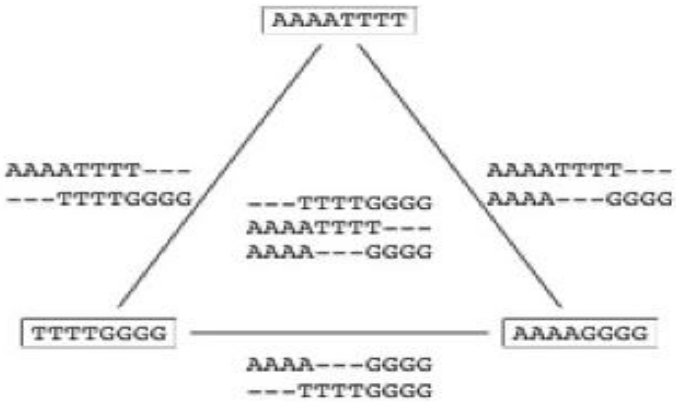
**can we construct a multiple alignment that induces them?**

**NOT ALWAYS**

**Pairwise alignments may be inconsistent**

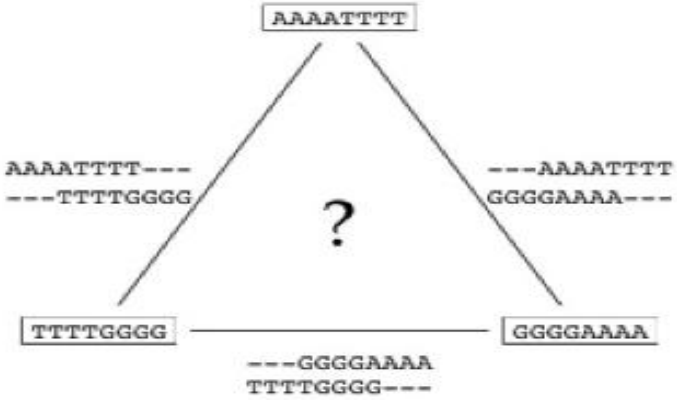
# Combining Optimal Pairwise Alignments into Multiple Alignment

Can combine pairwise alignments into multiple alignment




(a) Compatible pairwise alignments

Can *not* combine pairwise alignments into multiple alignment




(b) Incompatible pairwise alignments

# Multiple Alignment: Greedy Approach

- Choose most similar pair of strings and combine into a profile , thereby reducing alignment of *k* sequences to an alignment of of *k-1* sequences/profiles. **Repeat**
- This is a heuristic greedy method

$$k \left\{ \begin{array}{l} u_1 = \text{ACGTACGTACGT}\ldots \\ u_2 = \text{TTAATTAATTAA}\ldots \\ u_3 = \text{ACTACTACTACT}\ldots \\ \ldots \\ u_k = \text{CCGGCCGGCCGG} \end{array} \right. \longrightarrow \left. \begin{array}{l} u_1 = \text{ACg/tTACg/tTACg/cT}\ldots \\ u_2 = \text{TTAATTAATTAA}\ldots \\ \ldots \\ u_k = \text{CCGGCCGGCCGG}\ldots \end{array} \right\} k\text{-}1$$

# Greedy Approach: Example

- Consider these 4 sequences

| | |
|---|---|
| *s1* | GATTCA |
| *s2* | GTCTGA |
| *s3* | GATATT |
| *s4* | GTCAGC |

# Greedy Approach: Example (cont' d)

- There are $\binom{4}{2}$ = 6 possible alignments

```
s2  GTCTGA                  s1  GATTCA--
s4  GTCAGC (score = 2)      s4  G–T-CAGC(score = 0)

s1  GAT-TCA                 s2  G-TCTGA
s2  G-TCTGA (score = 1)     s3  GATAT-T (score = -1)

s1  GAT-TCA                 s3  GAT-ATT
s3  GATAT-T (score  = 1)    s4  G-TCAGC (score = -1)
```

# Greedy Approach: Example (cont'd)

$s_2$ *and* $s_4$ are closest; combine:

*s2* GTCTGA
*s4* GTCAGC

$s_{2,4}$ GTCt/aGa/c
(profile)

new set of 3 sequences:

$s_1$ GATTCA
$s_3$ GATATT
$s_{2,4}$ GTCt/aGa/c

# Progressive Alignment

- *Progressive alignment* is a variation of greedy algorithm with a somewhat more intelligent strategy for choosing the order of alignments.
- Progressive alignment works well for close sequences, but deteriorates for distant sequences
  - Gaps in consensus string are permanent
  - Use profiles to compare sequences

# ClustalW

- Popular multiple alignment tool today
- ‘W’ stands for ‘weighted’ (different parts of alignment are weighted differently).
- Three-step process

  1.) Construct pairwise alignments

  2.) Build Guide Tree

  3.) Progressive Alignment guided by the tree

# Step 1: Pairwise Alignment

- Aligns each sequence again each other giving a similarity matrix
- Similarity = exact matches / sequence length (percent identity)

| | $v_1$ | $v_2$ | $v_3$ | $v_4$ |
|---|---|---|---|---|
| $v_1$ | - | | | |
| $v_2$ | .17 | - | | |
| $v_3$ | .87 | .28 | - | |
| $v_4$ | .59 | .33 | .62 | - |

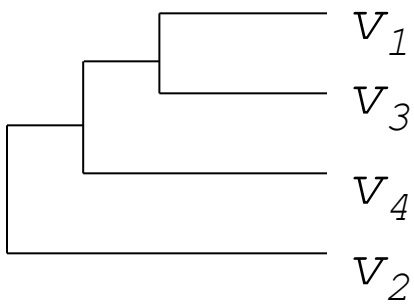
(.17 means 17 % identical)

# Step 2: Guide Tree

- Create Guide Tree using the similarity matrix
  - ClustalW uses the neighbor-joining method
  - Guide tree roughly reflects evolutionary relations

# Step 2: Guide Tree (cont' d)

| | $v_1$ | $v_2$ | $v_3$ | $v_4$ |
|---|---|---|---|---|
| $v_1$ | - | | | |
| $v_2$ | .17 | - | | |
| $v_3$ | .87 | .28 | - | |
| $v_4$ | .59 | .33 | .62 | - |




Calculate:

$v_{1,3}$ = alignment $(v_1, v_3)$

$v_{1,3,4}$ = alignment$((v_{1,3}), v_4)$

$v_{1,2,3,4}$ = alignment$((v_{1,3,4}), v_2)$

# Step 3: Progressive Alignment

- Start by aligning the two most similar sequences
- Following the guide tree, add in the next sequences, aligning to the existing alignment
- Insert gaps as necessary

```
FOS_RAT         PEEMSVTS-LDLTGGLPEATTPESEEAFTLPLLNDPEPK-PSLEPVKNISNMELKAEPFD
FOS_MOUSE       PEEMSVAS-LDLTGGLPEASTPESEEAFTLPLLNDPEPK-PSLEPVKSISNVELKAEPFD
FOS_CHICK       SEELAAATALDLG----APSPAAAEEAFALPLMTEAPPAVPPKEPSG--SGLELKAEPFD
FOSB_MOUSE      PGPGPLAEVRDLPG-----STSAKEDGFGWLLPPPPPPP-----------------LPFQ
FOSB_HUMAN      PGPGPLAEVRDLPG-----SAPAKEDGFSWLLPPPPPPP-----------------LPFQ
                .   . :   ** .     :..  *:.*   *   . *                  **:
```

Dots and stars show how well-conserved a column is.

# Multiple Alignments: Scoring

- Number of matches (multiple longest common subsequence score)
- Entropy score
- Sum of pairs (SP-Score)

# Multiple LCS Score

- A column is a “match” if all the letters in the column are the same

AAA
AAA
AAT
ATC

- Only good for very similar sequences

# Entropy

- Define frequencies for the occurrence of each letter in each column of multiple alignment
  - $p_A = 1$, $p_T = p_G = p_C = 0$ (1st column)
  - $p_A = 0.75$, $p_T = 0.25$, $p_G = p_C = 0$ (2nd column)
  - $p_A = 0.50$, $p_T = 0.25$, $p_C = 0.25$ $p_G = 0$ (3rd column)
- Compute entropy of each column

$$-\sum_{X=A,T,G,C} p_X \log p_X$$

AAA
AAA
AAT
ATC

# Entropy: Example

$$entropy\begin{pmatrix} A \\ A \\ A \\ A \end{pmatrix} = 0$$

Best case

Worst case

$$entropy\begin{pmatrix} A \\ T \\ G \\ C \end{pmatrix} = -\sum \frac{1}{4}\log\frac{1}{4} = -4(\frac{1}{4} * -2) = 2$$

# Multiple Alignment: Entropy Score

Entropy for a multiple alignment is the sum of entropies of its columns:

$$\Sigma_{\text{ over all columns }} \Sigma_{X=A,T,G,C}\, p_X \log p_X$$

# Entropy of an Alignment: Example

column entropy:

$$-( p_A \log p_A + p_C \log p_C + p_G \log p_G + p_T \log p_T)$$

| | | |
|---|---|---|
| A | A | A |
| A | C | C |
| A | C | G |
| A | C | T |

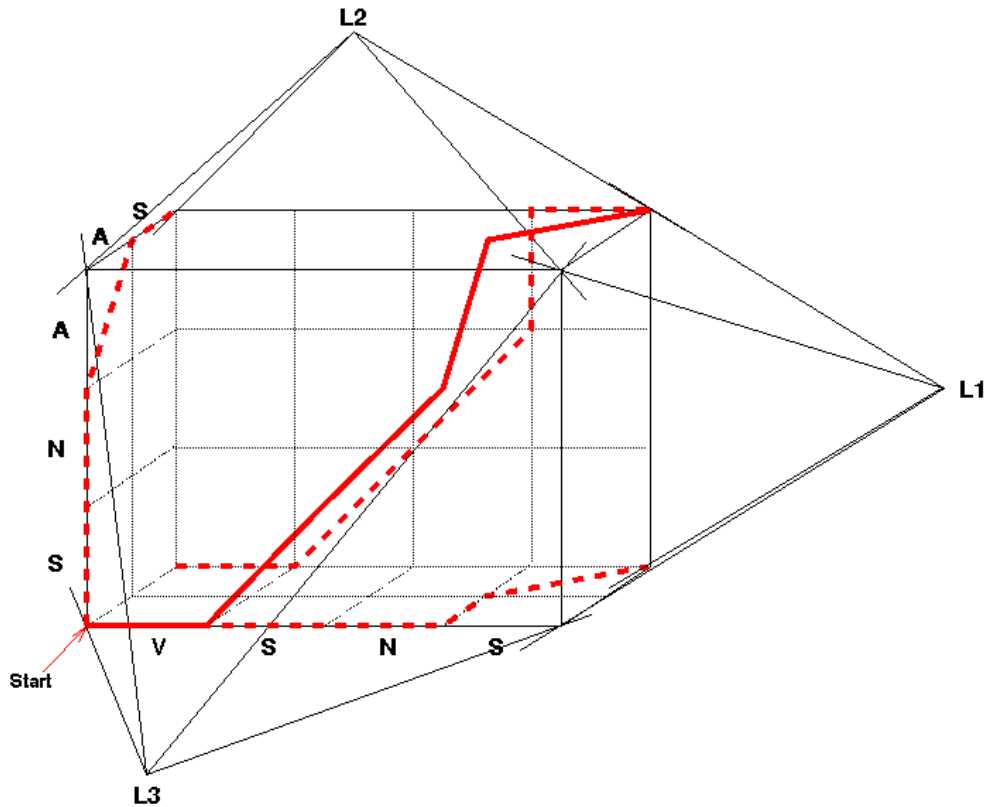
- Column 1 = -[1*log(1) + 0*log0 + 0*log0 +0*log0]
  = 0
- Column 2 = -[($^1/_4$)*log($^1/_4$) + ($^3/_4$)*log($^3/_4$) + 0*log0 + 0*log0]
  = -[ ($^1/_4$)*(-2) + ($^3/_4$)*(-.415) ] = +0.811
- Column 3 = -[($^1/_4$)*log($^1/_4$)+($^1/_4$)*log($^1/_4$)+($^1/_4$)*log($^1/_4$) +($^1/_4$) *log($^1/_4$)] = 4* -[($^1/_4$)*(-2)] = +2.0
- Alignment Entropy = 0 + 0.811 + 2.0 = +2.811

# Inferring Pairwise Alignments from Multiple Alignments

- From a multiple alignment, we can infer pairwise alignments between all sequences, but they are not necessarily optimal

- This is like projecting a 3-D multiple alignment path on to a 2-D face of the cube

# Multiple Alignment Projections




All 3 Pairwise Projections of the Multiple Alignment

A 3-D alignment can be projected onto the 2-D plane to represent an alignment between a pair of sequences.

# Sum of Pairs Score(SP-Score)

- Consider pairwise alignment of sequences

  $a_i$ and $a_j$

  imposed by a multiple alignment of $k$ sequences
- Denote the score of this suboptimal (not necessarily optimal) pairwise alignment as

  $s^*(a_i, a_j)$
- Sum up the pairwise scores for a multiple alignment:

$$s(a_1,\ldots,a_k) = \Sigma_{i,j}\, s^*(a_i, a_j)$$

# Computing SP-Score

Aligning 4 sequences: 6 pairwise alignments

Given $a_1, a_2, a_3, a_4$:

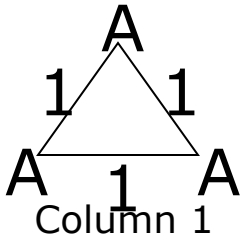
$$s(a_1 \ldots a_4) = \Sigma s^*(a_i, a_j) = s^*(a_1, a_2) + s^*(a_1, a_3) + s^*(a_1, a_4) + s^*(a_2, a_3) + s^*(a_2, a_4) + s^*(a_3, a_4)$$

# SP-Score: Example

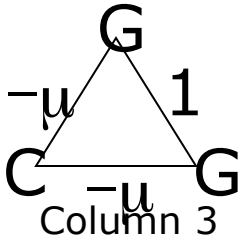
$a_1$ ATG–C–AAT
· A–G–CATAT
$a_k$ ATCCCATTT

To calculate each column:

$$s'(a_1 \ldots a_k) = \sum_{i,j} s^*(a_i, a_j) \longleftarrow \binom{n}{2} \text{Pairs of Sequences}$$




Score=3




Score = $1 - 2\mu$

# Multiple Alignment: History

***1975 Sankoff***

*Formulated multiple alignment problem and gave dynamic programming solution*

***1988 Carrillo-Lipman***

*Branch and Bound approach for MSA*

***1990 Feng-Doolittle***

*Progressive alignment*

***1994 Thompson-Higgins-Gibson-ClustalW***

*Most popular multiple alignment program*

***1998 Morgenstern et al.-DIALIGN***

*Segment-based multiple alignment*

***2000 Notredame-Higgins-Heringa-T-coffee***

*Using the library of pairwise alignments*

***2004 MUSCLE***

***What's next?***

# Problems with Multiple Alignment

- Multidomain proteins evolve not only through point mutations but also through domain duplications and domain recombinations
- Although MSA is a 30 year old problem, there were no MSA approaches for aligning **rearranged** sequences (i.e., multi-domain proteins with shuffled domains) prior to 2002
- Often impossible to align all protein sequences throughout their entire length

# Profile Representation of Multiple Alignment

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| - | A | G | G | C | T | A | T | C | A | C | C | T | G |
| T | A | G | - | C | T | A | C | C | A | - | - | - | G |
| C | A | G | - | C | T | A | C | C | A | - | - | - | G |
| C | A | G | - | C | T | A | T | C | A | C | - | G | G |
| C | A | G | - | C | T | A | T | C | G | C | - | G | G |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | | 1 | | | | 1 | | | .8 | | | | |
| C | .6 | | | 1 | | | .4 | 1 | | .6 | .2 | | |
| G | | | 1 | .2 | | | | | .2 | | | .4 | 1 |
| T | .2 | | | | 1 | | .6 | | | | | .2 | |
| - | .2 | | | .8 | | | | | | .4 | .8 | .4 | |

# Profile Representation of Multiple Alignment

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| - | A | G | G | C | T | A | T | C | A | C | C | T | G |
| T | A | G | - | C | T | A | C | C | A | - | - | - | G |
| C | A | G | - | C | T | A | C | C | A | - | - | - | G |
| C | A | G | - | C | T | A | T | C | A | C | - | G | G |
| C | A | G | - | C | T | A | T | C | G | C | - | G | G |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | | 1 | | | | 1 | | | .8 | | | | |
| C | .6 | | | 1 | | | .4 | 1 | | .6 | .2 | | |
| G | | | 1 | .2 | | | | | .2 | | | .4 | 1 |
| T | .2 | | | | 1 | | .6 | | | | | .2 | |
| - | .2 | | | .8 | | | | | | .4 | .8 | .4 | |

In the past we were aligning a **sequence against a sequence**

Can we align a **sequence against a profile?**

Can we align a **profile against a profile?**