
Multiple Sequence Alignment

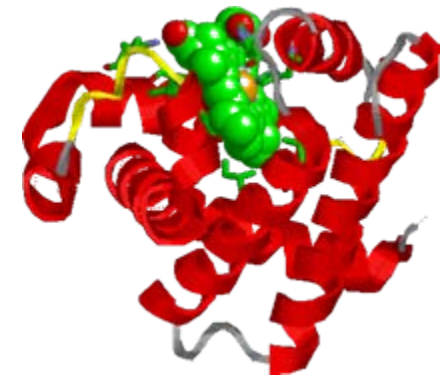
-supplemental material for CSE 200:
Scientific Computing in Matlab

Weixiong Zhang

Department of Computer Science and Engineering
Washington University in St. Louis

A Little Bit Biology

- **DNA** - double helix of 4 distinct molecules (nucleotides): A, C, T, G.
- **Proteins** –
 - ◆ long chains made from 20 different molecules, amino acids
 - ◆ main building blocks for life - 3D structures and functions.
- Genes in DNA sequences determines proteins
- Similarity in DNA sequences implies structural and functional similarities in proteins



Why Multiple Global Alignment – A Basic, Useful Tool

- **Given sequences of different species,**
 - ◆ Discover the **evolutionary history** by constructing phylogenetic trees
- **Given a new (protein) sequence,**
 - ◆ Identify the **family** it belongs to
 - ◆ Identify its **conserved biological features**
 - Predict protein 3D structures (related to protein folding)
 - Infer biological functions
- **A basic tool for sequence assembly**

What is Multiple Global Alignment

(1) A C G A G A
(2) A T A A T G
(3) C T G T G

(1) A C G A G A
(2) A T A A T G
(3) C T G T G

$$4 + 2 + 5 + 2 + 5 + 4 + 2 = 24$$

Scoring function:

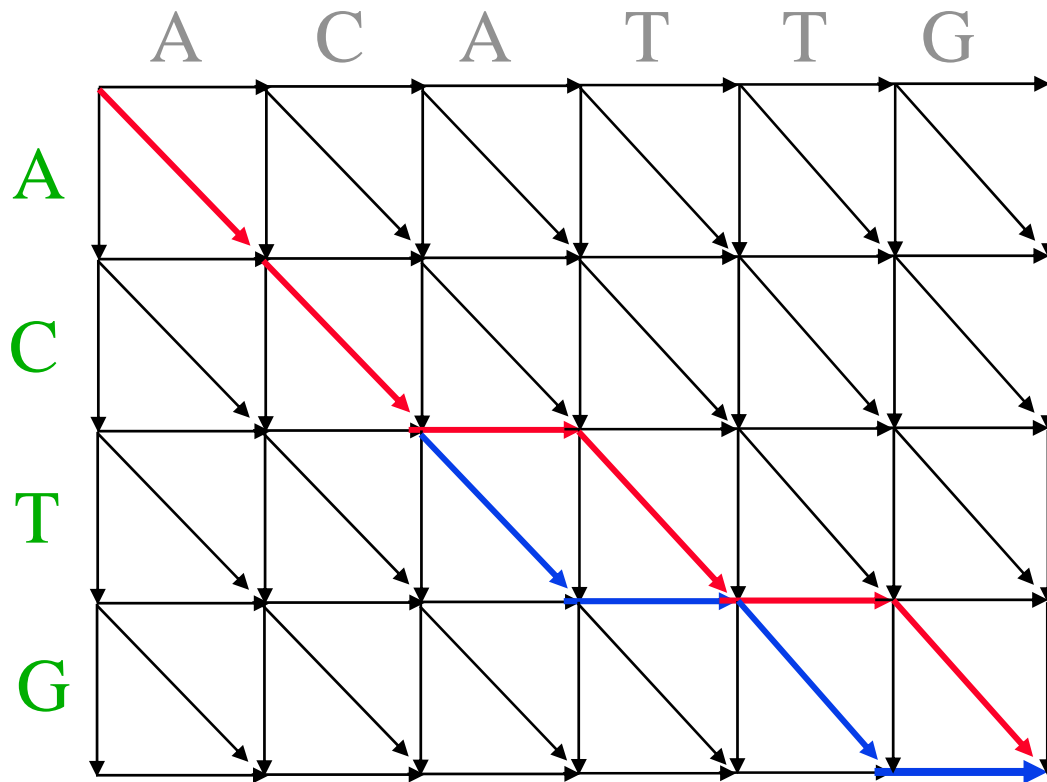
A – A = 0 (match)

A – G = 1 (substitute)

A – = 2 (insert or deletion - indel)

- Sum-of-Pairs (SP) cost function

Alignment as Path Finding – 2D Case

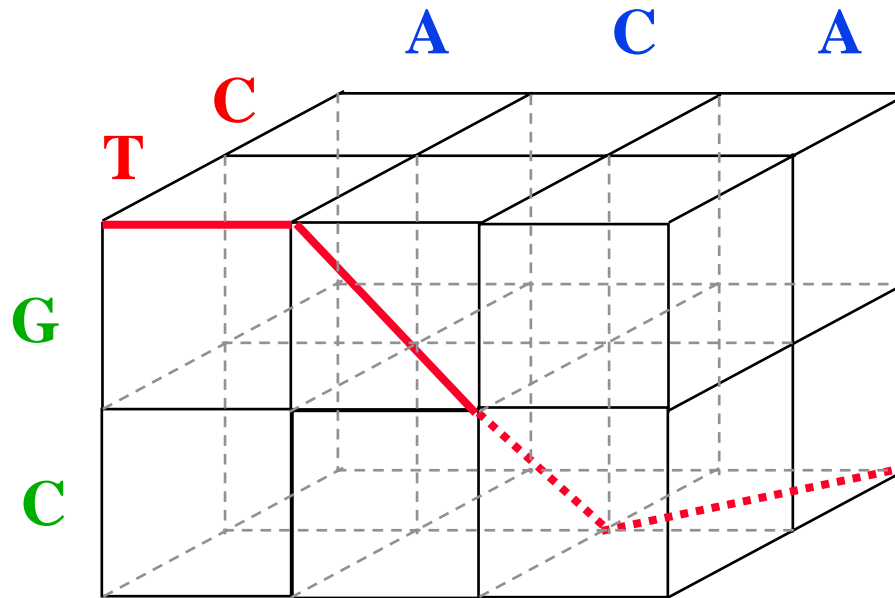


$$\begin{array}{r}
 A C A T T G \\
 A C \quad T \quad G \\
 \hline
 0 \ 0 \ 2 \ 0 \ 2 \ 0 = 4
 \end{array}$$

$$\begin{array}{r}
 A C A T T G \\
 A C T \quad G \\
 \hline
 0 \ 0 \ 1 \ 2 \ 1 \ 2 = 6
 \end{array}$$

A
A

Alignment as Path Finding – 3D Case



(1) A C A

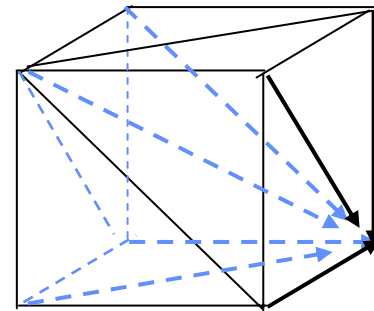
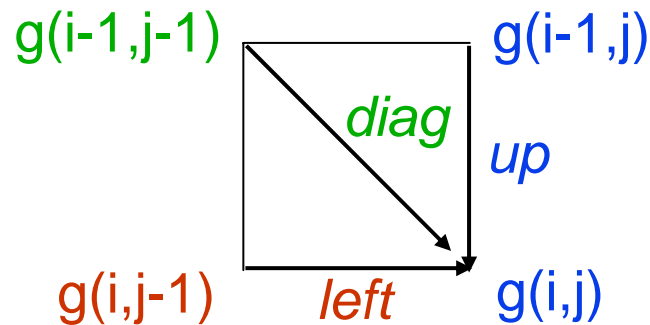
(2) T C

(3) G C

Local recursion

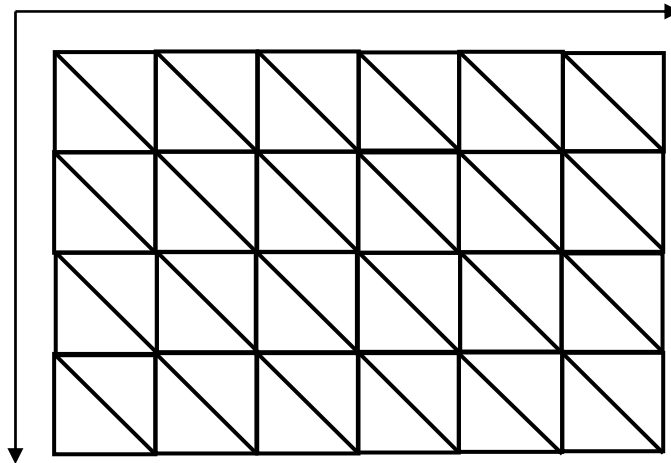
- **Recursive relation** [Needleman&Wunsch, 1970]

- ◆ $g(i, j)$: optimal path cost from (0,0) to (i, j)
- ◆ $g(i, j) = \min\{g(i-1, j) + \text{up}; g(i, j-1) + \text{left}; g(i-1, j-1) + \text{diag}\}$
- ◆ Calculate $g(i, j)$ row by row, or column by column



Problem formulation

- ◆ From the left to the right, or from the top to the bottom

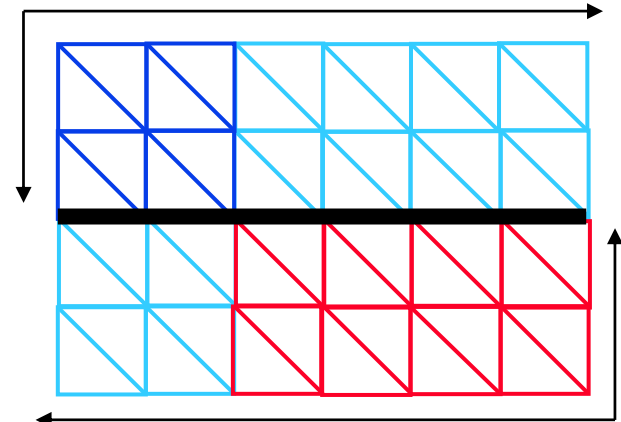
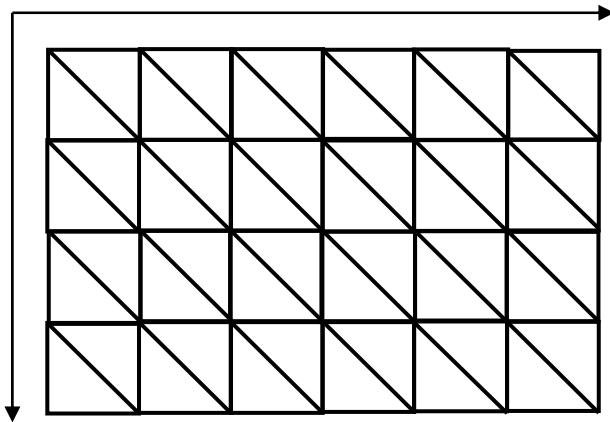


- Save all cells to find the shortest path
 - Need $O(n^d)$ space, where n is the length of a sequence and d the number of sequences. $O(n^2)$ for 2 sequences

Divide-and-conquer

- **Divide-and-Conquer**: [Hirschberg, 1975; Myers&Miller, 1988]

- ◆ reduce memory from $O(n^d)$ to $O(n^{d-1})$, where n is the length of a sequence and d the number of sequences.



- ◆ **Overhead on computation:** $r = N(\text{new})/N(\text{old}) = d/(d-1)$
 - $r = 2$ ($d=2$)
 - $r \rightarrow 1$ ($d \rightarrow \text{infinity}$)
 - $r = 3$, $n = 1000$, one byte per base. $n^3 = 1\text{GB}$, $n^2 = 1\text{MB}$