

Optimisation Alignment. (60 minutes)

<http://www.stats.ox.ac.uk/~hein/lectures.htm>

α -globin (141) and β -globin (146)

V-LSPADKTNVKAANGKVGAGHAGEYGAELERMFLSFPTTKTYFPHF-DLS--H---GSAQVKGHGKKVADAL
VHLTPEEKSAVTALWGKV--NVDEVGGEALGRLLVVYPWTQRFFESFGDLSTPDAVMGNPKVKAHGKKVLGAF
TNAVAHVDDMPNALSALSSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTISKYR
SDGLAHL DNLKGT FATLSELHCDKLHVDPENFRLLGNVLVCVLAHHFGKEFTPPVQAAYQKVVAGVANALAHKYH

It often matches functional region with functional region

Determines homology at residue/nucleotide level.

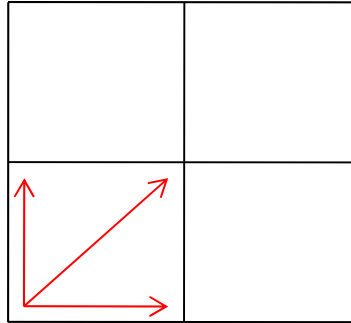
Similarity/Distance between molecules can be evaluated

Molecular Evolution studies.

Homology/Non-homology depends on it.

Number of alignments, $T(n,m)$

Alignments columns are equivalent to step (0,1), (1,0) and (1,1) in a $[0,n][0,m]$ matrix.



$T(n,m)$ is the number of alignments of $s1[1,n]$ and $s2[1,m]$ then

$$T(n,m) = T(n-1,m) + T(n,m-1) + T(n-1,m-1)$$

$$T(0,0) = 1 \quad T(n,m) > 3^{\min(n,m)}$$

Thus alignment by alignment search for best alignment is not realistic.

If $\binom{n-}{-n}$ is equivalent to $\binom{-n}{n-}$

then alignments are equivalent to choosing two subsets of $s1$ and $s2$ that has to be matched, thus

$$T(n,m) = \sum_{i=1}^{\min(n,m)} \binom{n}{i} \binom{m}{i}$$

		1	9	41	129	321	681
T		1	7	25	63	129	231
G		1	5	13	25	41	61
T		1	3	5	7	9	11
F		1	1	1	1	1	1
		C	T	A	G	G	

Parsimony Alignment of two strings.

Sequences: s1=CTAGG s2=TTGT. 5, indels (g) 10.

Basic operations:

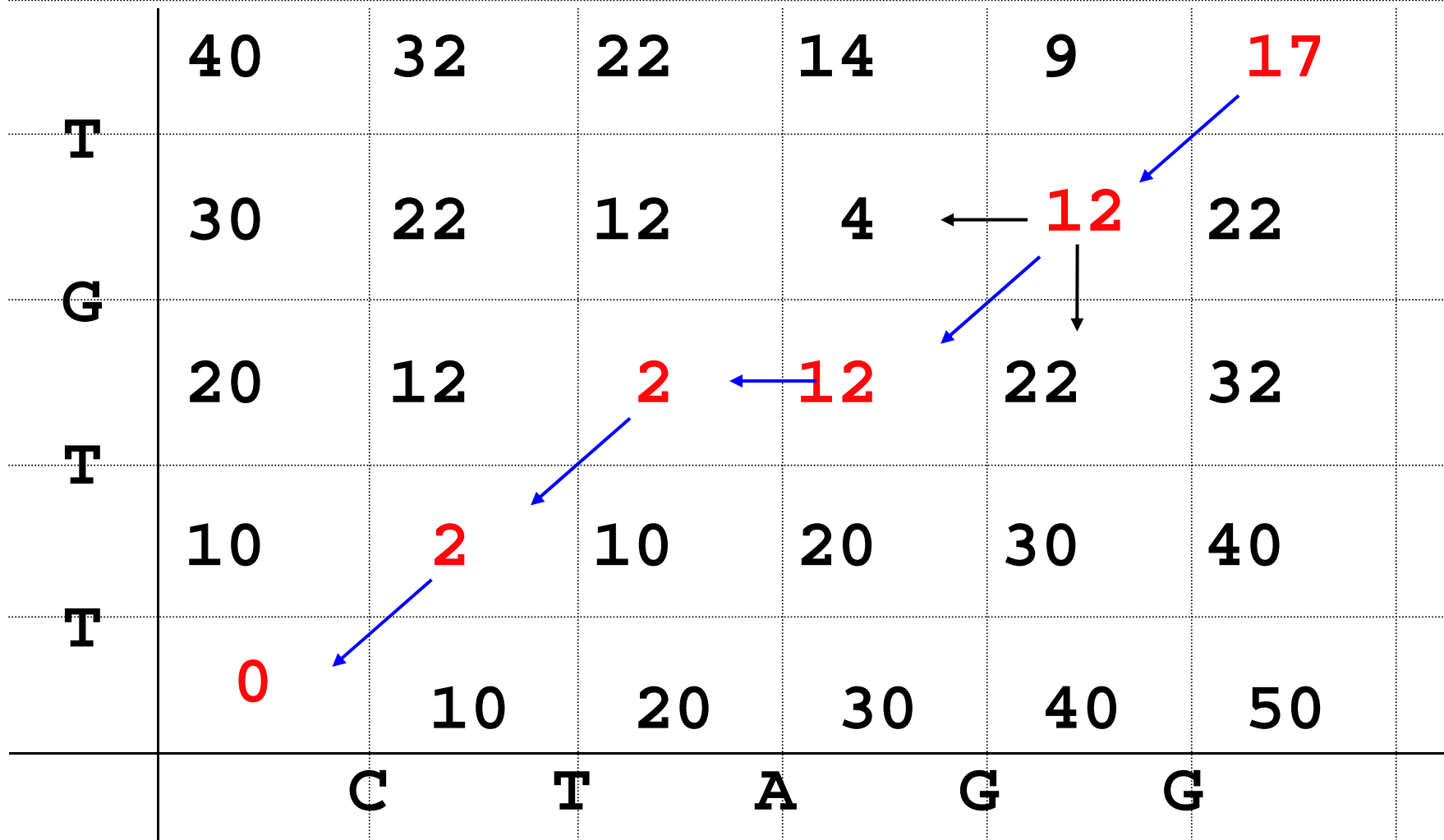
transitions 2 (C-T & A-G), transversions 5, indels (g) 10.

$$\text{Cost Additivity} \quad \begin{array}{c} \text{CTAG} \\ \text{TT-G} \end{array} = \begin{array}{c} \text{CTA} \\ \text{TT-} \end{array} + \begin{array}{c} \text{G} \\ \text{G} \end{array}$$

$$\begin{array}{l} \{CTAG, TTG\}_{AL} \\ 12 \end{array} = \text{Min} \left[\begin{array}{l} \{CTA, TT\}_{AL} + GG \\ 12 \quad 0 \end{array} \quad (A) \right. \\ \left. \begin{array}{l} \{CTA, TTG\}_{AL} + G- \\ 4 \quad 10 \end{array} \quad (B) \right. \\ \left. \begin{array}{l} \{CTAG, TT\}_{AL} + -G \\ 32 \quad 10 \end{array} \quad (C) \right]$$

$$D_{i,j} = \min\{D_{i-1,j-1} + d(s1[i], s2[j]), D_{i,j-1} + g, D_{i-1,j} + g\}$$

Initial condition: $D_{0,0}=0$. ($D_{i,j} := D(s1[1:i], s2[1:j])$)



Alignment :

CTAGG

i v

Cost 17

TT-GT

Complexity of Accelerations of pairwise algorithm.

Dynamical Programming: $(n+1)(m+1)3=O(nm)$

Backtracking: $O(n+m)$

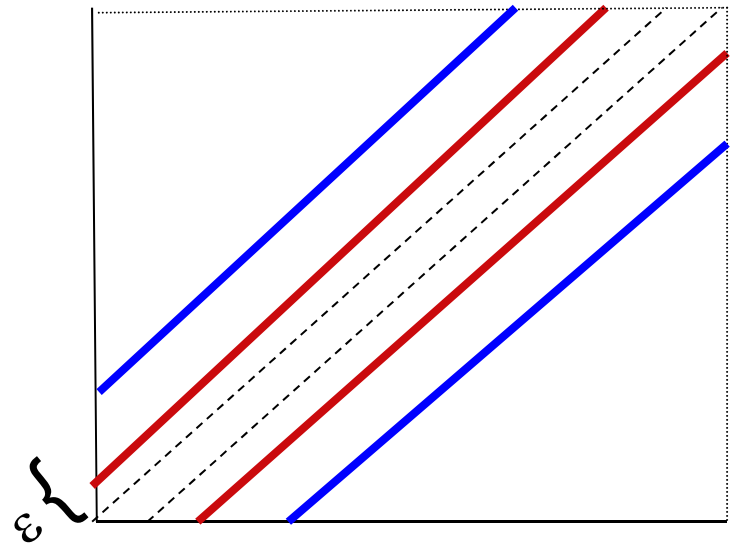
Recursion without memory: $T(n,m) > 3^{\min(n,m)}$

Exact acceleration (Ukkonen, Myers).

Assume all events cost 1.

If $d_\varepsilon(s_1, s_2) < 2\varepsilon + |l_1 - l_2|$, then

$d(s_1, s_2) = d_\varepsilon(s_1, s_2)$



Heuristic acceleration: Smaller band & larger acceleration, but no guarantee of optimum.

Close-to-Optimum Alignments

(Waterman & Byers, 1983)

Alignments within ϵ of optimal

Ex. $\epsilon = 2$.

	40	32	22	14	9	*	17
T				*	/		
	30	22	12	4	12	22	
G			*	/			
	20	12	2	-	12	22	32
T		/					
	10	2	10	20	30	40	
T	/						
	0	10	20	30	40	50	
	C	T	A	G	G		

C	T	A	G	G
i		i	v	g
T	T	G	T	-

Cost 19

Caveat:

There are enormous numbers of suboptimal alignments.

Hirschberg & Close-to-Optimum Alignments

(Hirschberg, 1975).

Sets of positions that are on some suboptimal alignment.

Alignments within ϵ of optimal. Ex. $\epsilon = 2$

	40/50	32/40	22/30	14/20	9/10	17/0
T						
	30/40	22/30	12/25	4/15	12/5	22/10
G						
	20/35	12/25	2/15	<u>12/5</u>	22/10	32/20
T						
	10/25	2/15	10/15	20/15	30/20	40/30
T						
	0/17	10/15	20/20	30/25	40/30	50/40
	C	T	A	G	G	

Mid point: (3,2) and the alignment problem is then reduced to 2 smaller alignment problems: (CTA + TT) and (GG + GT)

Longer Indels

TCATGGTACCGTTAGCGT
GCA-----GCAT

g_k : cost of indel of length k (for instance $10 + \log k$)

$$D_{i,j} = \min \{$$

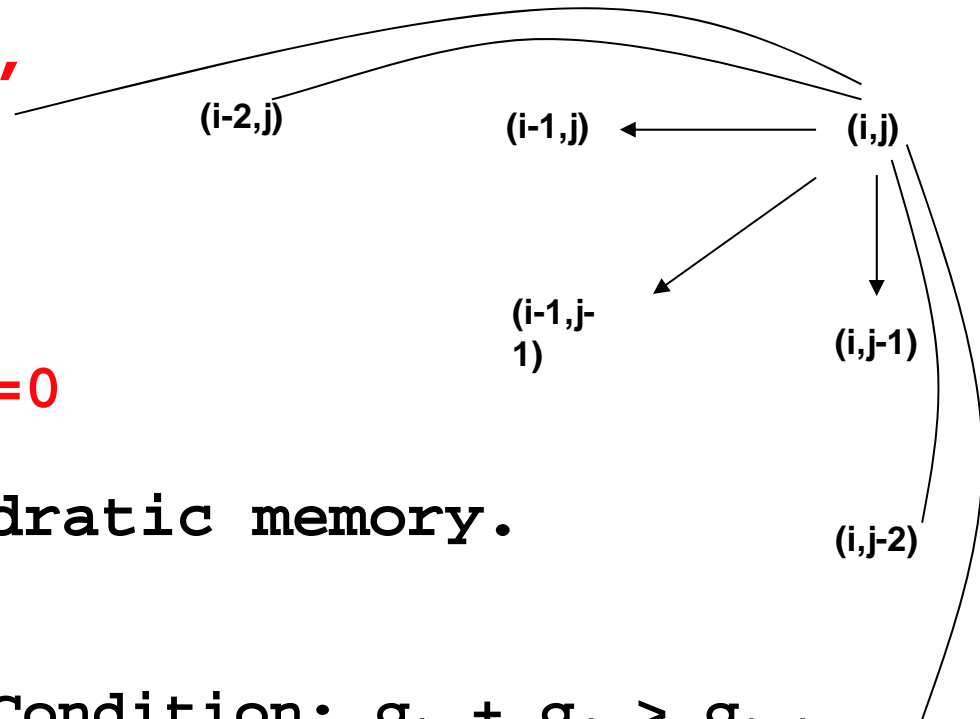
$$\begin{aligned} & D_{i-1,j-1} + d(s1[i],s2[j]), \\ & D_{i,j-1} + g_1, D_{i,j-2} + g_2, \\ & D_{i-1,j} + g_1, D_{i-2,j} + g_2, \\ & \} \end{aligned}$$

Initial condition: $D_{0,0}=0$

Cubic running time. Quadratic memory.

Comment:

Evolutionary Consistency Condition: $g_i + g_j > g_{i+j}$



Distance-Similarity

(Smith-Waterman-Fitch, 1982)

$$S_{i,j} = \max\{S_{i-1,j-1} + s(s1[i],s2[j]), S_{i,j-1} - w, S_{i-1,j} - w\}$$

Similarity
 $s(n1,n2)$
 w

Distance
 $M - d(n1,n2)$
 $1/(2*M) + g$

Similarity: Transversions:0 Transitions:3 Identity:5 Indels: 10 + 1/10

Distance: Transitions:2 Transversions 5 Identity 0 Indels:10. M largest dist (5)

T	40/-40.4	32/-27.3	22/-12.2	14/0.9	9/11.0	17/2.9
G	30/-30.3	22/-17.2	12/-2.1	4/11.0	12/2.9	22/-7.2
T	20/-20.2	12/-7.1	2/8.0	12/-2.1	22/-12.2	32/-22.3
T	10/-10.1	2/3.0	10/-7.1	20/-17.2	30/-27.3	40/-37.4
	0/0	10/-10.1	20/-20.2	30/-30.3	40/-40.4	50/-50.5
	C	T	A	G	G	

1. The Switch from Dist to Sim is highly analogous to Maximizing $\{-f(x)\}$ instead of Minimizing $\{f(x)\}$.

2. Dist will be based on a metric:

- i. $d(x,x) = 0$, ii. $d(x,y) \geq 0$, iii. $d(x,y) = d(y,x)$ &
- iv. $d(x,z) + d(z,y) \geq d(x,y)$.

There are no analogous restrictions on Sim, giving it a larger parameter space.

Local alignment

Smith, Waterman (1981)

Global Alignment:

$$S_{i,j} = \max\{D_{i-1,j-1} + s(s1[i],s2[j]), S_{i,j-1} - w, S_{i-1,j} - w\}$$

Local:

$$S_{i,j} = \max\{D_{i-1,j-1} + s(s1[i],s2[j]), S_{i,j-1} - w, S_{i-1,j} - w, 0\}$$

	0	1	0	.6	1	2	.6	1.6	1.6	3	2.6
C	0	0	1	0	1	.3	.6	0.6	2	3	1.6
A	0	0	0	1.3	0	1	1	2	<u>3.3</u>	2	1.6
G	0	0	.3	.3	1.3	1	2.3	<u>2.3</u>	2	.6	1.6
C	0	0	.6	1.6	.3	1.3	<u>2.6</u>	2.3	1	.6	1.6
U	0	0	2	.6	.3	<u>1.6</u>	2.6	1.3	1	.6	1
A	0	1	.6	0	1	<u>3</u>	1.6	1.3	1	1.3	1.6
C	0	1	0	0	<u>2</u>	1.3	.3	1	.3	2	.6
C	0	0	0	<u>1</u>	.3	0	0	.6	1	0	0
G	0	0	<u>0</u>	.6	1	0	0	0	1	1	2
U	0	0	1	.6	0	0	0	0	0	0	0
A	0	0	1	0	0	0	0	0	0	0	0
A	0	0	0	0	0	0	0	0	0	0	0
	C	A	G	C	C	U	C	G	C	U	U

Score Parameters:

Match: 1

Mismatch -1/3

Gap 1 + k/3

GCC-UCG

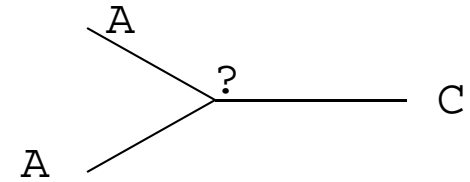
GCCAUG

Alignment of three sequences.

s1=ATCG s2=ATGCC s3=CTCC

Alignment: AT-CG
 ATGCC
 CT-CC

A
A
C



Consensus sequence: ATCC

Configurations in an alignment column:

-	-	n	n	n	-	n	-
-	n	-	n	-	n	n	-
n	-	-	-	n	n	n	-

Recursion: $D_{i,j,k} = \min\{D_{i-i',j-j',k-k'} + d(i,i',j,j',k,k')\}$

Initial condition: $D_{0,0,0} = 0.$

Running time: $l_1 * l_2 * l_3 * (2^3 - 1)$ Memory requirement: $l_1 * l_2 * l_3$

New phenomena: ancestral/consensus sequence.

Parsimony Alignment of four sequences

s1=ATCG s2=ATGCC s3=CTCC s4=ACGCG

Alignment: AT-CG G
 ATGCC C
 CT-CC C
 ACGCG G



Configurations in alignment columns:

-	-	-	n	-	-	-	n	n	n	-	n	n	n	n	-
-	-	n	-	n	n	-	n	-	-	n	-	n	n	n	-
-	n	-	-	n	-	n	-	n	-	n	n	-	n	n	-
n	-	-	-	-	n	n	-	-	n	n	n	n	-	n	-

Recursion: $D_i = \min\{D_{i-\Delta} + d(i, \Delta)\} \quad \Delta \in [\{0,1\}^4 \setminus \{0\}^4]$

Initial condition: $D_0 = 0$. Memory : $l_1 * l_2 * l_3 * l_4$

Computation time: $l_1 * l_2 * l_3 * l_4 * 2^4$ Memory : $l_1 * l_2 * l_3 * l_4$

New Phenomena: Cost and alignment is phylogeny dependent

Alignment of many sequences.

s1=ATCG, s2=ATGCC,, sn=ACGCG



Configurations in an alignment column: $2^n - 1$
 \in

Recursion: $D_i = \min\{D_{i-\Delta} + d(i, \Delta)\} \Delta \in [\{0, 1\}^n \setminus \{0\}^n]$

Initial condition: $D_{0,0,\dots,0} = 0.$

Computation time: $l^n * (2^n - 1) * n$ Memory requirement: l^n
(l: sequence length, n: number of sequences)

Progressive Alignment

(Feng-Doolittle 1987 J.Mol.Evol.)

Can align alignments and given a tree make a multiple alignment.

```

*
alkmny-trwq
akkmdyftrwq
kkkmemftrwq

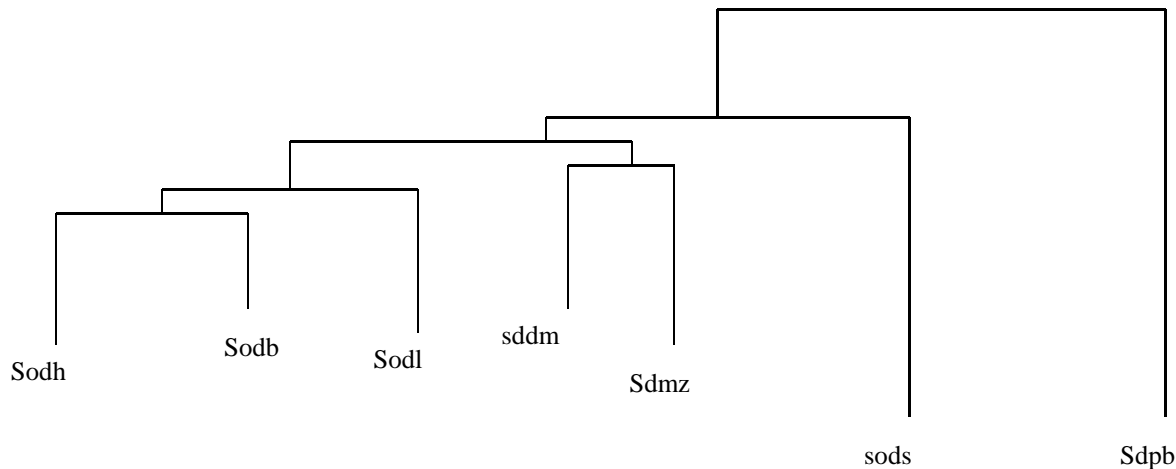
```

```

*
acdeqrt
acdehrt

```

[P(n,q) + P(n,h) + P(d,q) + P(d,h) + P(e,q) + P(e,h)] / 6



* * * * *

```

Sodh  atkavcvlkgdgpqvqgsinfeqkesdgpvkvwgsikglte-ghgfhvhqfg----ndtagct  sagphfnp lsrk
Sodb  atkavcvlkgdgpqvqgtinfeak-gdtvkvwgsikglte--ghgfhvhqfg----ndtagct  sagphfnp lsrk
Sodl  atkavcvlkgdgpqvqgsinfeqkesdgpvkvwgsikglte-ghgfhvhqfg----ndtagct  sagphfnp lsrk
Sddm  atkavcvlkgdgpqvq -infeak-gdtvkvwgsikglte--ghgfhvhqfg----ndtagct  sagphfnp lsrk
Sdmz  atkavcvlkgdgpqvq- infeqkesdgpvkvwgsikglte-ghgfhvhqfg----ndtagct  sagphfnp Lsrk
Sods  vatkavcvlkgdgpqvq- infek-gdtvkvwgsikgltepnglhgfhvhqfg----ndtagct  sagphfnp lsrk
Sdpb  datkavcvlkgdgpqvq--infeqkesdgpv---wgsikgltglhgfhvhqfgscasndtagctvlggssagphfnpehtnk

```

Summary

Comparison of 2 Strings

- *Minimize Distance-Maximize Similarity*
- *Dynamical Programming Algorithm*
- *Local alignment*
- *Close-to-Optimal Solutions*

Comparison of many Strings

- *Simultaneous Phylogeny and Alignment*