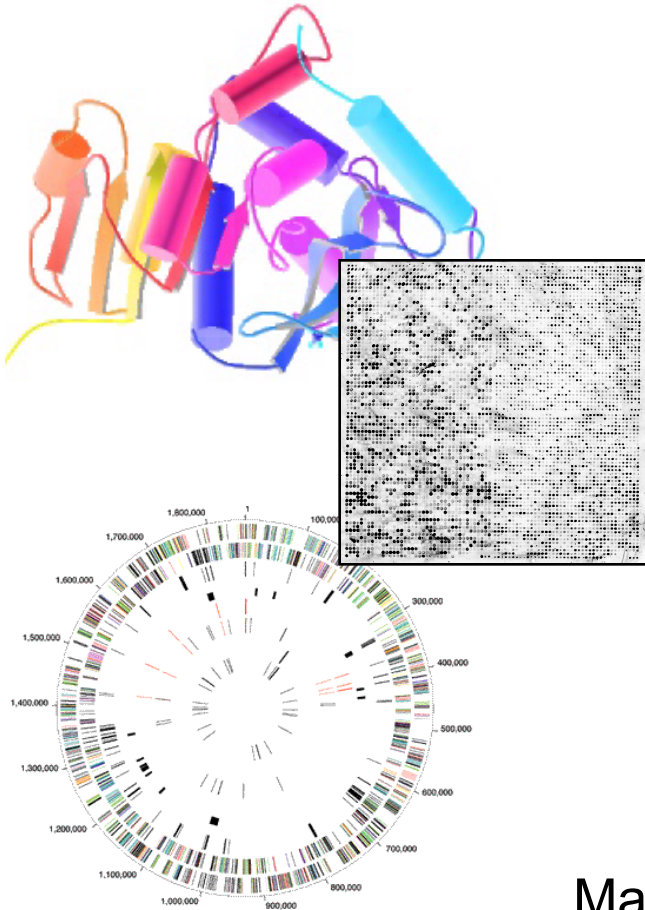


BIOINFORMATICS

Sequence Comparison



Mark Gerstein, Yale University
GersteinLab.org/courses/452
(last edit in spring '17, MG lecture #2, final edit)

Sequence Topics (Contents)

- Basic Alignment via Dynamic Programming
- Suboptimal Alignment
- Gap Penalties
- Similarity (PAM) Matrices
- Local Alignment

Aligning Text Strings

Raw Data ???

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

2 matches, 0 gaps

```
T C A T G
      | |
C A T T G
```

3 matches (2 end gaps)

```
T C A T G .
  | | |
. C A T T G
```

4 matches, 1 insertion

```
T C A - T G
      | |   | |
. C A T T G
```

4 matches, 1 insertion

```
T C A T - G
      | | |   |
. C A T T G
```

Dynamic Programming

- What to do for a bigger string?

SSDSEREEHVKRFRQALDDTGMKVPMATNLFTHPVFKDGGFTANDRDVRRYALRKTIRNIDLAVELGAETYVAWGGREGAESGGAKDVRDALDRMKEAFDLLGEYVTSQGYDIRFAIEP
KPNEPRGDILLPTVGHALAFIERLERPELYGVNPEVGHEQMAGLNFPHGIAQALWAGKLFHIDLNGQNGIKYDQDLRFGAGDLRAAFWLVDLLESAGYSGPRHFDKPPRTEDFDGVWAS

- Needleman-Wunsch (1970) provided first automatic method

- ◇ Dynamic Programming to Find Global Alignment

- Their Test Data (J → Y)

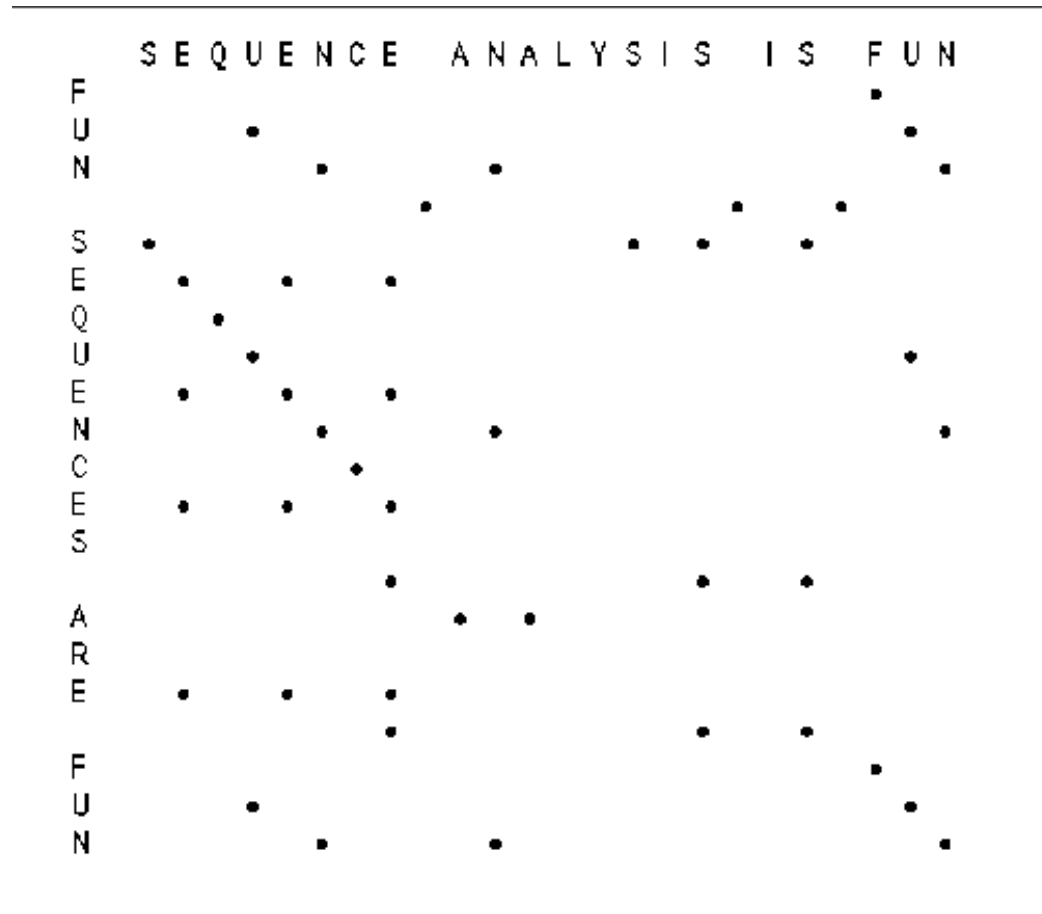
- ◇ ABCNYRQCLCRPM
AYCYNRCKCRBP

Step 1 -- Make a Dot Plot (Similarity Matrix)

Put 1's where characters are identical.

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	1												
Y					1								
C			1					1		1			
Y					1								
N				1									
R						1					1		
C			1					1		1			
K													
C			1					1		1			
R						1					1		
B		1											
P												1	

A More Interesting Dot Matrix



(adapted from R Altman)

Step 2 -- Start Computing the Sum Matrix

```

new_value_cell(R,C) <=
  cell(R,C)           { Old value, either 1 or 0   }
  + Max[
    cell (R+1, C+1),   { Diagonally Down, no gaps   }
    cells(R+1, C+2 to C_max), { Down a row, making col. gap }
    cells(R+2 to R_max, C+1) { Down a col., making row gap }
  ]

```

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	1												
Y					1								
C			1					1		1			
Y					1								
N				1									
R						1					1		
C			1					1		1			
K													
C			1					1		1			
R						1					1		
B		1											
P												1	

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	1												
Y					1								
C			1					1		1			
Y					1								
N				1									
R						1					1		
C			1					1		1			
K													
C			1					1		1			
R						1					2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0



Step 2 -- Start Computing the Sum Matrix

```

new_value_cell(R,C) <=
  cell(R,C)           { Old value, either 1 or 0   }
  + Max[
    cell (R+1, C+1),   { Diagonally Down, no gaps   }
    cells(R+1, C+2 to C_max), { Down a row, making col. gap }
    cells(R+2 to R_max, C+1) { Down a col., making row gap }
  ]
  
```

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	1												
Y					1								
C			1					1		1			
Y					1								
N				1									
R						1					1		
C			1					1		1			
K													
C			1					1		1			
R						1					1		
B		1											
P												1	

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	1												
Y					1								
C			1					1		1			
Y					1								
N				1									
R						1					1		
C			1					1		1			
K													
C			1					1		1			
R						1					2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

Step 3 -- Keep Going

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	1												
Y					1								
C			1					1		1			
Y					1								
N				1									
R						1					1		
C			1					1		1			
K													
C			1					1		1			
R						1					2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	1												
Y					1								
C			1					1		1			
Y					1								
N				1									
R						5	4	3	3	2	2	0	0
C	3	3	4	3	3	3	3	4	3	3	1	0	0
K	3	3	3	3	3	3	3	3	3	2	1	0	0
C	2	2	3	2	2	2	2	3	2	3	1	0	0
R	2	1	1	1	1	2	1	1	1	1	2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0



Step 4 -- Sum Matrix All Done

Alignment Score is 8 matches.

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	1												
Y					1								
C			1					1		1			
Y					1								
N				1									
R						5	4	3	3	2	2	0	0
C	3	3	4	3	3	3	3	4	3	3	1	0	0
K	3	3	3	3	3	3	3	3	3	2	1	0	0
C	2	2	3	2	2	2	2	3	2	3	1	0	0
R	2	1	1	1	1	2	1	1	1	1	2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	8	7	6	6	5	4	4	3	3	2	1	0	0
Y	7	7	6	6	6	4	4	3	3	2	1	0	0
C	6	6	7	6	5	4	4	4	3	3	1	0	0
Y	6	6	6	5	6	4	4	3	3	2	1	0	0
N	5	5	5	6	5	4	4	3	3	2	1	0	0
R	4	4	4	4	4	5	4	3	3	2	2	0	0
C	3	3	4	3	3	3	3	4	3	3	1	0	0
K	3	3	3	3	3	3	3	3	3	2	1	0	0
C	2	2	3	2	2	2	2	3	2	3	1	0	0
R	2	1	1	1	1	2	1	1	1	1	2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0



Step 5 -- Traceback

Find Best Score (8) and Trace Back

A B C N Y - R Q C L C R - P M
 A Y C - Y N R - C K C R B P

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	8	7	6	6	5	4	4	3	3	2	1	0	0
Y	7	7	6	6	6	4	4	3	3	2	1	0	0
C	6	6	7	6	5	4	4	4	3	3	1	0	0
Y	6	6	6	5	6	4	4	3	3	2	1	0	0
N	5	5	5	6	5	4	4	3	3	2	1	0	0
R	4	4	4	4	4	5	4	3	3	2	2	0	0
C	3	3	4	3	3	3	3	4	3	3	1	0	0
K	3	3	3	3	3	3	3	3	3	2	1	0	0
C	2	2	3	2	2	2	2	3	2	3	1	0	0
R	2	1	1	1	1	2	1	1	1	1	2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

Hansel & Gretel

Step 6 -- Alternate Tracebacks

A B C - N Y R Q C L C R - P M
 A Y C Y N - R - C K C R B P

Also,
 Suboptimal
 Alignments

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	8	7	6	6	5	4	4	3	3	2	1	0	0
Y	7	7	6	6	6	4	4	3	3	2	1	0	0
C	6	6	7	6	5	4	4	4	3	3	1	0	0
Y	6	6	6	5	6	4	4	3	3	2	1	0	0
N	5	5	5	6	5	4	4	3	3	2	1	0	0
R	4	4	4	4	4	5	4	3	3	2	2	0	0
C	3	3	4	3	3	3	3	4	3	3	1	0	0
K	3	3	3	3	3	3	3	3	3	2	1	0	0
C	2	2	3	2	2	2	2	3	2	3	1	0	0
R	2	1	1	1	1	2	1	1	1	1	2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

Gap Penalties

The score at a position can also factor in a penalty for introducing gaps (i. e., not going from i, j to $i-1, j-1$).

Gap penalties are often of linear form:

$$\text{GAP} = a + bN$$

GAP is the gap penalty

a = cost of opening a gap

b = cost of extending the gap by one (affine)

N = length of the gap

(Here assume $b=0$, $a=1/2$, so $\text{GAP} = 1/2$ regardless of length.)

ATGCAAAT

ATG-AAAAT .5

ATG--AAAT .5 + (1)b [b=.1]

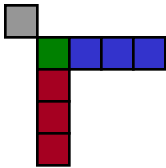
ATG---AAT .5 + (2)(.1) = .7

Step 2 -- Computing the Sum

Matrix with Gaps

```

new_value_cell(R,C) <=
  cell(R,C)           { Old value, either 1 or 0 }
  + Max[
    cell(R+1,C+1),    { Diagonally Down, no gaps }
    cells(R+1,C+2 to C_max) - GAP, { Down a row, making col. gap }
    cells(R+2 to R_max,C+1) - GAP { Down a col., making row gap }
  ]
  
```



	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	1												
Y					1								
C			1					1	1				
Y					1								
N				1									
R						1					1		
C			1					1	1				
K													
C			1					1	1				
R						1					1		
B		1											
P												1	

	A	B	C	N	Y	R	Q	C	L	C	R	P	M
A	1												
Y					1								
C			1					1	1				
Y					1								
N				1									
R						1					1		
C			1					1	1				
K													
C			1					1	1				
R						1					1.5	0	0
B	1	1.5	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

GAP
 = 1/2

All Steps in Aligning a 4-mer

C R B P

C R P M

- C R P M

C R - P M

	C	R	P	M
C	1			
R		1		
B				
P			1	

	C	R	P	M
C	1			
R		2	0	0
B	1	1	0	0
P	0	0	1	0

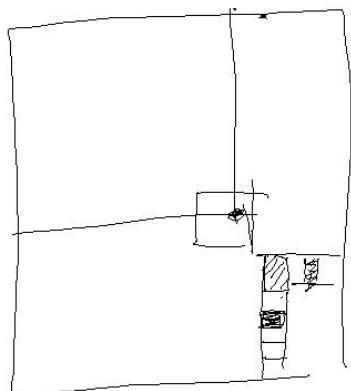
	C	R	P	M
C	3	1	0	0
R	1	2	0	0
B	1	1	0	0
P	0	0	1	0

	C	R	P	M
C	3	1	0	0
R	1	2	0	0
B	1	1	0	0
P	0	0	1	0

Bottom right hand corner of previous matrices

Key Idea in Dynamic Programming

- ◇ The best alignment that ends at a given pair of positions (i and j) in the 2 sequences is the score of the best alignment previous to this position PLUS the score for aligning those two positions.
- ◇ An Example Below
 - Aligning R to K does not affect alignment of previous N-terminal residues. Once this is done it is **fixed**. Then go on to align D to E.
 - How could this be violated?
Aligning R to K changes best alignment in box.



ACSQRP--LRV-SH	R SENCV
A-SNKPQLVKLMTH	V K DFCV

ACSQRP--LRV-SH	-R	S E NCV
A-SNKPQLVKLMTH	VK	D FCV

Similarity (Substitution)

Matrix

- Identity Matrix

- ◊ Match L with L => 1
- Match L with D => 0
- Match L with V => 0??

- S(aa-1,aa-2)

- ◊ Match L with L => 1
- Match L with D => 0
- Match L with V => .5

- Number of Common Ones

- ◊ PAM
- ◊ Blossum
- ◊ Gonnet

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	4	-1	-2	-2	0	-1	-1	0	-2	-1	-1	-1	-1	-2	-1	1	0	-3	-2	0
R	-1	5	0	-2	-3	1	0	-2	0	-3	-2	2	-1	-3	-2	-1	-1	-3	-2	-3
N	-2	0	6	1	-3	0	0	0	1	-3	-3	0	-2	-3	-2	1	0	-4	-2	-3
D	-2	-2	1	6	-3	0	2	-1	-1	-3	-4	-1	-3	-3	-1	0	-1	-4	-3	-3
C	0	-3	-3	-3	8	-3	-4	-3	-3	-1	-1	-3	-1	-2	-3	-1	-1	-2	-2	-1
Q	-1	1	0	0	-3	5	2	-2	0	-3	-2	1	0	-3	-1	0	-1	-2	-1	-2
E	-1	0	0	2	-4	2	5	-2	0	-3	-3	1	-2	-3	-1	0	-1	-3	-2	-2
G	0	-2	0	-1	-3	-2	-2	6	-2	-4	-4	-2	-3	-3	-2	0	-2	-2	-3	-3
H	-2	0	1	-1	-3	0	0	-2	7	-3	-3	-1	-2	-1	-2	-1	-2	-2	2	-3
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4	2	-3	1	0	-3	-2	-1	-3	-1	3
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4	-2	2	0	-3	-2	-1	-2	-1	1
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5	-1	-3	-1	0	-1	-3	-2	-2
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5	0	-2	-1	-1	-1	-1	1
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6	-4	-2	-2	1	3	-1
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	6	-1	-1	-4	-3	-2
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4	1	-3	-2	-2
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5	-2	-2	0
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	10	2	-3
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	6	-1
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

Where do matrices come from?

- + \rightarrow More likely than random
- 0 \rightarrow At random base rate
- \rightarrow Less likely than random

- 1 Manually align protein structures
(or, more risky, sequences)
- 2 Look at frequency of a.a. substitutions
at structurally constant sites. -- i.e. pair i-j
exchanges

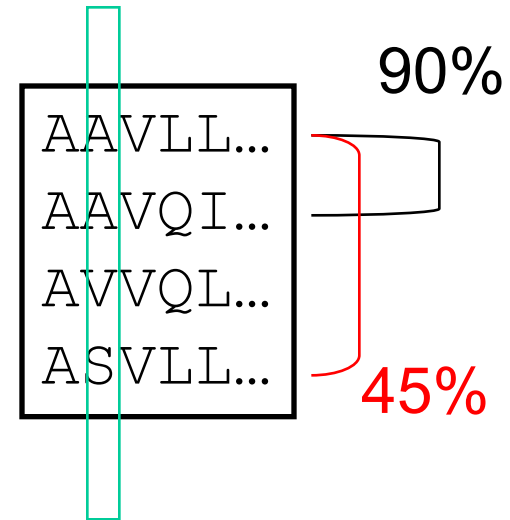
- 3 Compute log-odds

$$S(\text{aa-1}, \text{aa-2}) = \log_2 (\text{freq}(O) / \text{freq}(E))$$

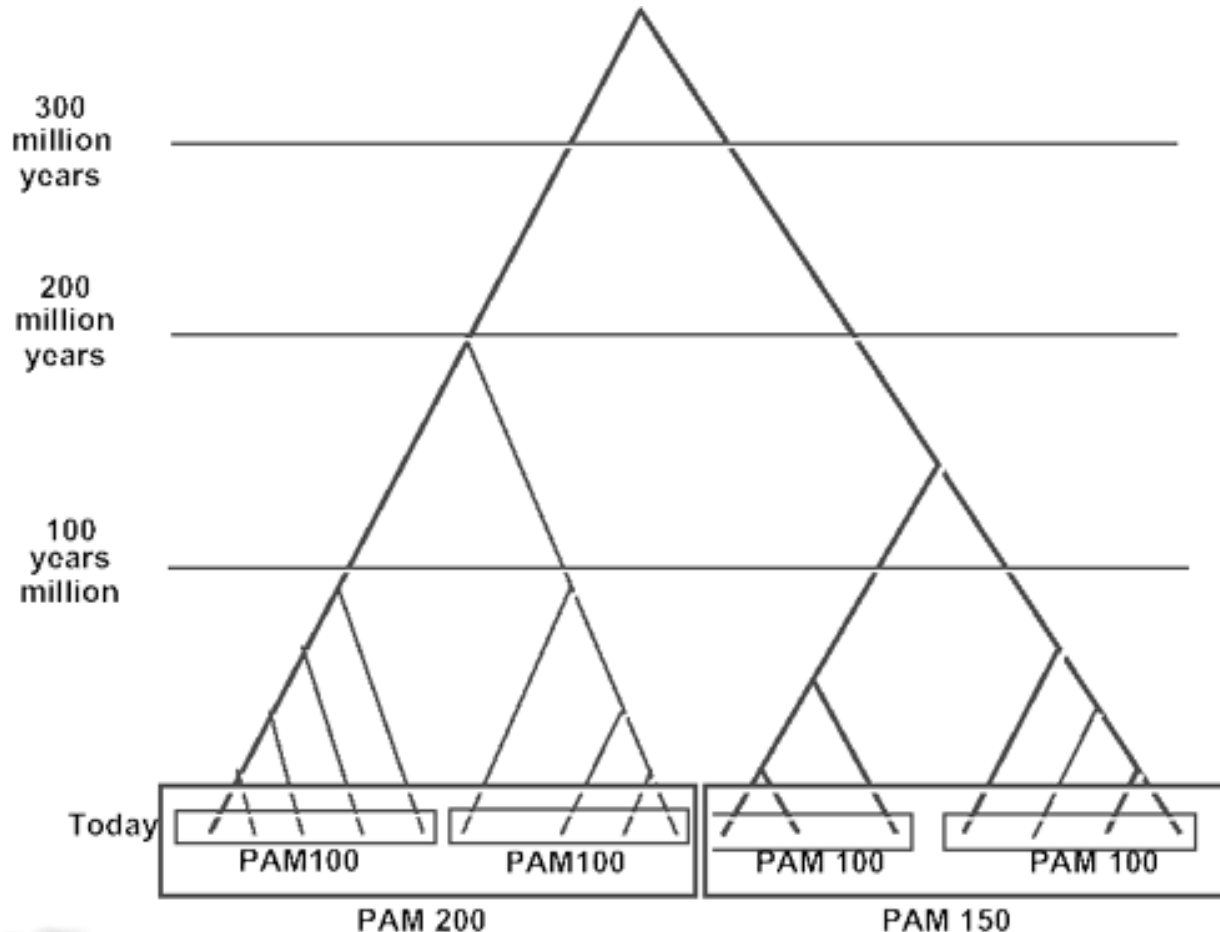
O = observed exchanges,

E = expected exchanges

- odds = freq(observed) / freq(expected)
- $S_{ij} = \log \text{ odds}$
- $\text{freq}(\text{expected}) = f(i) * f(j)$
= is the chance of getting amino acid i in a
column and then having it change to j
- e.g. A-R pair observed only a tenth as often as
expected



Different Matrices are Appropriate at Different Evolutionary Distances



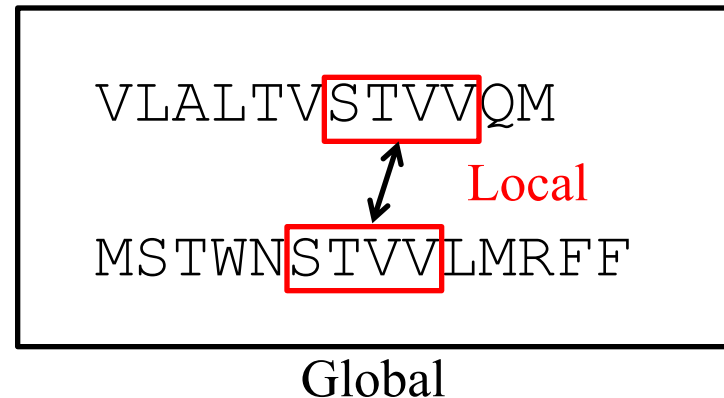
Different gold std. sets of seq at diff ev. dist. --> matrices

Ev. Equiv. seq. (ortholog) [hb and mb]

(Adapted from D Brutlag, Stanford)

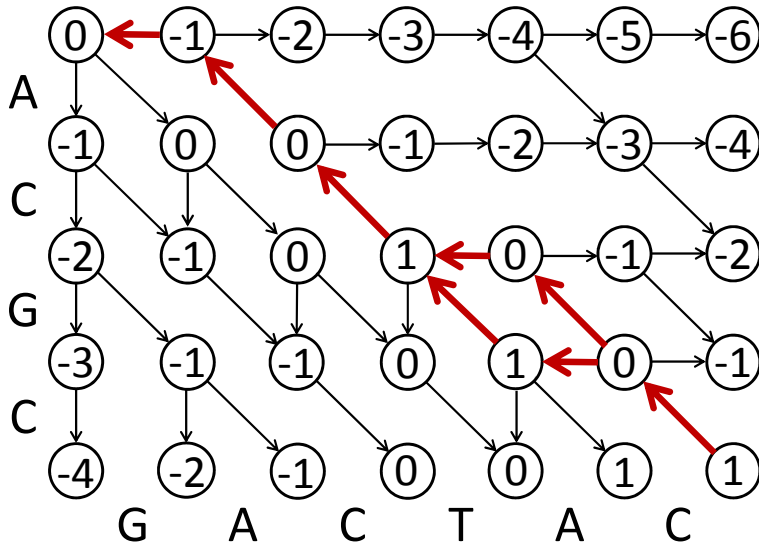
Modifications for Local Alignment

- 1 The scoring system uses negative scores for mismatches
 - 2 The minimum score for at a matrix element is zero
 - 3 Find the best score anywhere in the matrix (not just last column or row)
- These three changes cause the algorithm to seek high scoring subsequences, which are not penalized for their global effects (mod. 1), which don't include areas of poor match (mod. 2), and which can occur anywhere (mod. 3)



(Adapted from R Altman)

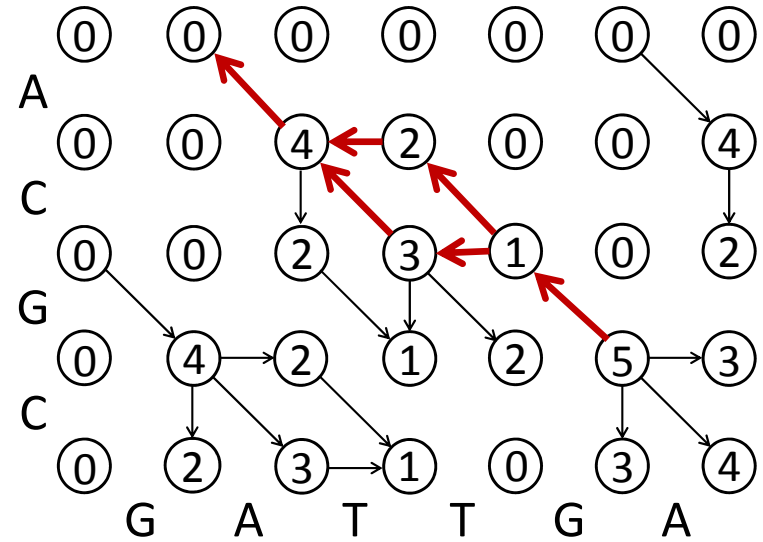
Global (NW) Vs. Local (SW) Alignments



-ACG-C
 || |
 GACTAC

or

-AC-GC
 || |
 GACTAC



AC-G
 | |
 GATTGA

or

A-CG
 | |
 GATTGA

Global : Match: +1 Mismatch: 0 Gap: -1

Local : Match: +4 Mismatch: -1 Gap: -2

(Adapted from Stephen F. Altschul)