Week 2
26 – 30 August

- ***Should have read***
  - Tree thinking
  - Dotplots, Ch 4.2
  - Ch 4.1-4.4
- ***Monday - Alignments/Dynamic Programming***
  - Ch 5.2, pgs 127 – 139
  - Read 5.1-5.2, 4.5
- ***Wednesday - Alignments /Scoring Systems***
  - Ch 4.3,5.1
- ***Friday – Alignments /Scoring Systems***

Sequence Comparison

Measuring the difference between sequences

```
ACGGTTAGCAAA      ACGGTTAGCAAA      ACGGTTAGCAAA
|||||||  ||||     ||||  ||  ||||     ||   |||  ||||
ACGGTTACCAAA      ACGGATACCAAA      ACCCTTACCAAA
```

1	2	3	Distance	Match Score
11	10	9	Similarity	

```
ACGGTTAGCAAA      ACGGTTAGCAAA      ACGGTTAGCAAA      ACGGTTAGCAAA
 |          ||     |   |    |||       |    |||          |||  ||||
CCTTACCAAAAC      CCTTACCAAAAC      CCTTACCAAAAC      CCTTACCAAAAC
```

Sequences may have to be offset to optimize score

Sequence Comparison

Measuring the difference between sequences

- ***What if sequences need spaces, what should the score be?***

```
AGTTACGGCAAA      AGTTACGGCAAA        AGTTACGGCAAA
|||||     |             |||||         |||||   |||||
AGTTAGCAAA          AGTTAGCAAACC      AGTTA..GCAAACC
```

```
ATCTAGCAG.T.C.A
  ||  |  ||  |  |  |
  CT.G.AGCTCCCA
```

- Allowing gaps makes it easier to get a high score.
- Intuitively, there should be some negative score for gaps.
- Otherwise any pair of random sequences can get a high score.

Sequence Comparison

Measuring the difference between sequences

- *Sequences alignments use a scoring function based on the number of matches and mismatches, and a function based on the number of gaps*

$$Match = N_{match} - N_{mismatch} - f(gap)$$

- *The score than unrelated sequences might get (on average) also matters*

Sequence Comparison

Finding the best alignment

- ***Without gaps – just slide the two sequences past each other and choose the offset with the highest score***
  - Requires time proportional to the square of the length of the sequences ($O(L^2)$)
- ***With gaps***
  - For each offset,
    - for each possible gap position
    - For each possible gap length
    - For each possible number of gaps
    - Calculate score ($O(L^L)$)

Sequence Comparison

Dynamic Programming Alignment

- ***Dynamic programming allows an optimal (highest scoring) alignment that considers all possible numbers and lengths of gaps to be found in $O(L^2)$ time***

- ***Dynamic programming uses a recursive definition of an optimal alignment***

- ***Alignment is guaranteed to be "optimal"***
  - Given: the scoring systems used and gap penalties

- ***Don't confuse optimal with correct - Even unrelated sequences can be optimally aligned!***

Sequence Comparison

Dynamic Programming Alignment – Recursive

- **Assume this alignment is optimal:**

```
AATGC
|  ||
AG.GC
```

- **If we remove the last base pair, like so**

```
AATG      C
|  |  +   |
AG.G      C
```

C:C must be the best pair that could be added, or it couldn't be optimal

- **The remaining part on the left HAS to be optimal**

Sequence Comparison

Dynamic Programming Alignment – Recursive

- *What if the remainder isn't optimal?*

```
AATG      C
|  |  +  |
AG.G      C
```

- *Suppose that there was something better*

```
AATG
| :|
A.GG
```

- *The original would not be optimal, there would be a better one*

```
AATGC
|  ||
AG.GC
```

```
AATGC
| :||
A.GGC
```

Sequence Comparison

Dynamic Programming Alignment – Recursive

- ***Remove one more pair***

```
AAT       G       C
|     +   |   +   |
AG.       G       C
```

- ***The remainder on the left has to be optimal***

- ***Look at this another way***
  - We can tell what the optimal alignment containing the terminal G:G and C:C pairs is, IF we know the optimal alignment up to that point
  - That is, if we know the optimal alignment between AAT and AG

Sequence Comparison

Dynamic Programming Alignment – Recursive

- *What are the possibilities?*

```
AAT      AAT      AAT
|        |         |
AG.      A.G      .AG
```

- *Alignment must use the next position in one of the sequences because we can't align gaps with gaps*
- *If we can't enumerate the possibilities, we can remove another position*

Sequence Comparison

Dynamic Programming Alignment – Recursive

- *Pick the best possible alignment of AAT and AG and then add the pairs we removed back*

```
AAT       G       C             AATGC
|     +   |   +   |     ->      |  ||
AG.       G       C             AG.GC
```

The best alignment containing G:G and C:C

- *If we did the same thing without removing the G:G pair, it would be the best alignment ending in C:C*

Sequence Comparison

Dynamic Programming Alignment

- ***Dynamic programming alignments look at every possible terminal pair and recursively calculates the optimal alignment***


```
AATG  C     AAT GC     AA  TGC     A   ATGC         AATGC
AGG   C     AGG C.     AGG C..     AGG C...     AGG C....

AATG C     AATG C.    AATG C..    AATG C...
AGG  C     AG   GC    A    GGC         AGGC
```

All possible ending pairs

Sequence Comparison

Dynamic Programming Alignment

- ***Alignment always uses the next character in one of the sequences at each position***

```
SEQUENCEONE
SEC.ONDLINE
```

- ***Some alignments may not be allowed***
  - Alignments with spaces in both sequences
    Can always add any number of positions with gap aligned with gap

```
SEQ.UENCEONE
SEC..ONDLINE
```

  - Alternating gaps

```
SEQ.UENC.E.ONE
SE.C.ON.D.LINE
```

```
S.E.Q.U.E.N.C.E.O.N.E
.S.E.C.O.N.D.L.I.N.E.
```

```
SEQUENCEONE..........
..........SECONDLINE
```

Sequence Comparison

Dynamic Programming

- ***Basic Recursion***
  The is the score, $S_{i,j}$, for the alignment ending with character i aligned with character j
  where $s_{i,j}$ is the score for matching character i and j
  and g is a gap penalty

$$S_{i,j} = \max\begin{Bmatrix} S_{i-1,j-1} + s_{i,j} \\ S_{i-1,j} + g \\ S_{i,j-1} + g \end{Bmatrix}$$ Eq 5.17

Sequence Comparison

Dynamic Programming Alignment

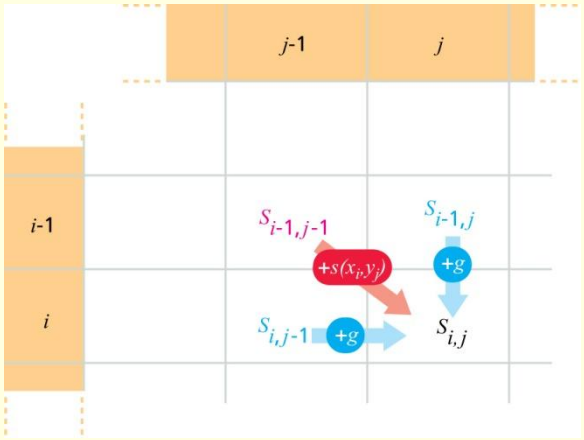
- ***A simple example: alignment of AATGC and AGGC***
- ***Score:***
  - Match = +1
  - Mismatch = 0
  - Gaps = -1
- ***Global alignment (all sequence characters used from both sequences)***

Sequence Comparison

Dynamic Programming Alignment

$$S_{i,j} = \max \begin{Bmatrix} S_{i-1,j-1} + s_{i,j} \\ S_{i-1,j} + g \\ S_{i,j-1} + g \end{Bmatrix}$$

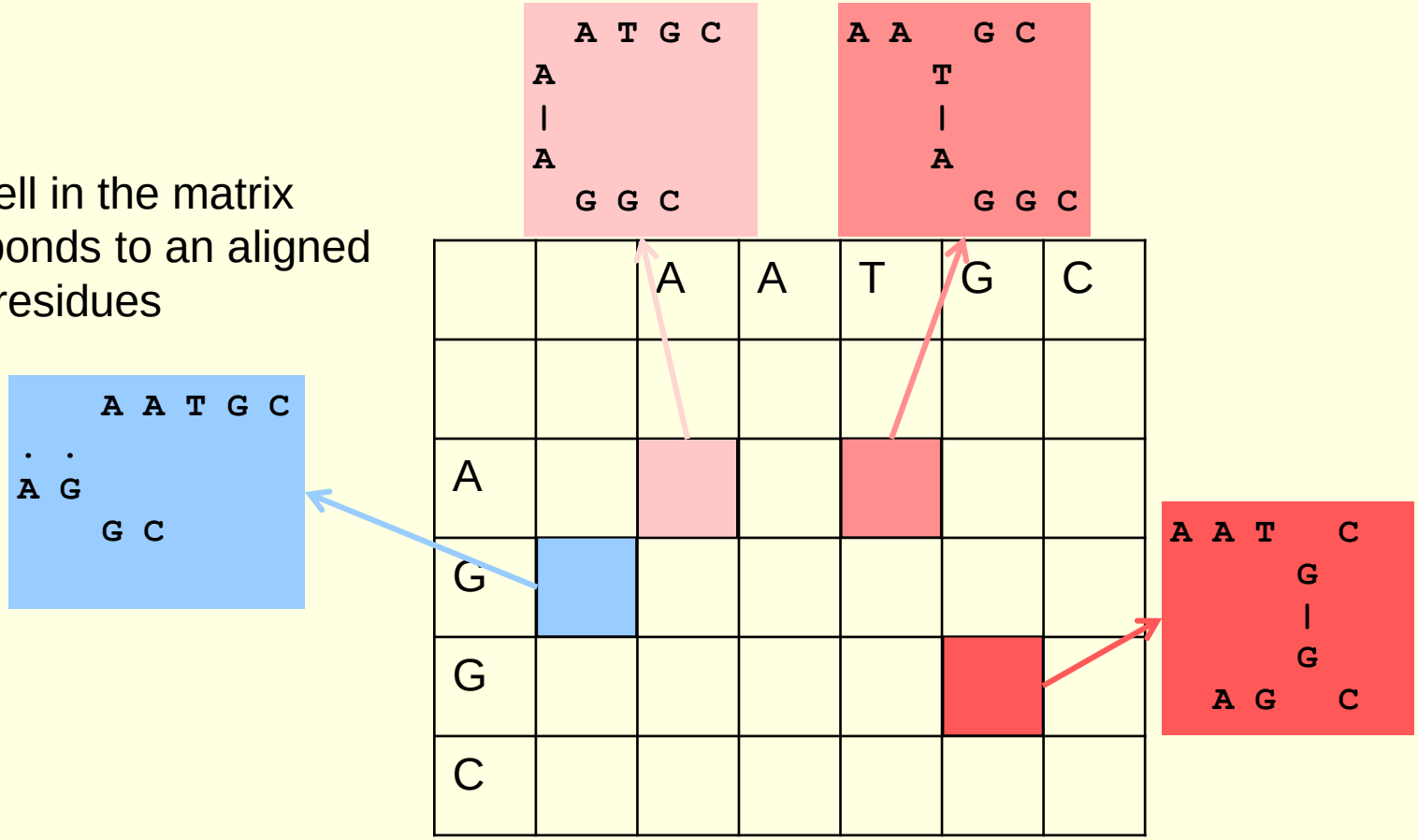
	A	A	T	G	C
A					
G					
G					
C					



Sequence Comparison

Dynamic Programming Alignment

Each cell in the matrix corresponds to an aligned pair of residues



Sequence Comparison

Dynamic Programming Alignment

		A	A	T	G	C
	0	→ -1	→ -2	→ -3	→ -4	→ -5
A	↓ -1	0+1 ↘ 1				
G	↓ -2					
G	↓ -3					
C	↓ -4					

Sequence Comparison

Dynamic Programming Alignment

		A	A	T	G	C
	0	-1	-2	-3	-4	-5
A	-1	1	0	1	2	3
G	-2					
G	-3					
C	-4					

Sequence Comparison

Dynamic Programming Alignment

		A	A	T	G	C
	0	-1	-2	-3	-4	-5
A	-1	0+1 1	-1+1 -1-1 0	-2+0 0-1 1	-3+0 -1-1 2	-4+0 -2-1 3
G	-2	-1+0 1-1 0	1+0 0-1 1	0+0 1-1 0	-1+1 0-1 -0	-2+0 -1-1 1
G	-3					
C	-4					

Sequence Comparison

Dynamic Programming Alignment

		A	A	T	G	C
	0	-1	-2	-3	-4	-5
A	-1	0+1 1	-1+1 1-1 0	-2+0 0-1 1	-3+0 1-1 2	-4+0 2-1 3
G	-2	-1+0 1-1 0	1+0 0-1 1	0+0 1-1 0	-1+1 0-1 0	-2+0 0-1 1
G	-3	-2+0 0-1 -1	0+0 1-1 0	1+0 0-1 1	0+1 -1-1 1-1 1	0+0 1-1 0
C	-4	-3+0 -1-1 -2	-1+0 0-1 -1	0+0 1-1 0 -1-1	1+0 1-1 1 0-1	1+1 0-1 2 1-1

Sequence Comparison

Dynamic Programming Alignment

		A	A	T	G	C
	0	-1	-2	-3	-4	-5
A	-1	0+1 1	-1+1 1-1 0	-2+0 -1	-3+0 -2	-4+0 -3
G	-2	-1+0 1-1 0	1+0 0-1 1	0+0 -2-1 1-1 0	-2+1-3-1 0-1 0	-3+0-4-1 -1-1 1
G	-3	-2+0 0-1 -1	0+0 1-1 0	1+0 0-1 0-1 1	0+1 -1-1 1-1 1	-1+0-2-1 1-1 0
C	-4	-3+0-1-1 -2	-1+0 0-1 -1	0+0 1-1 -1-1 0	1+0 1-1 0-1 1	1+1 1-1 2

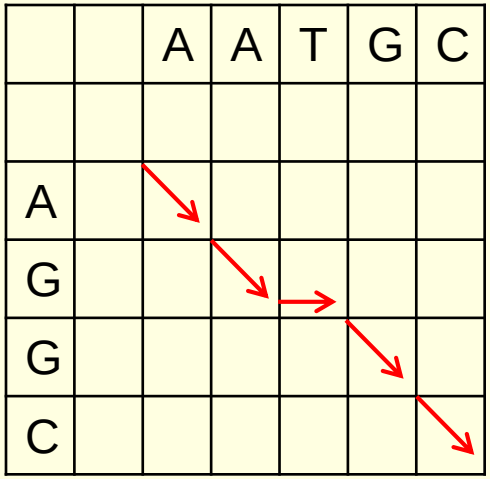
Score Matrix

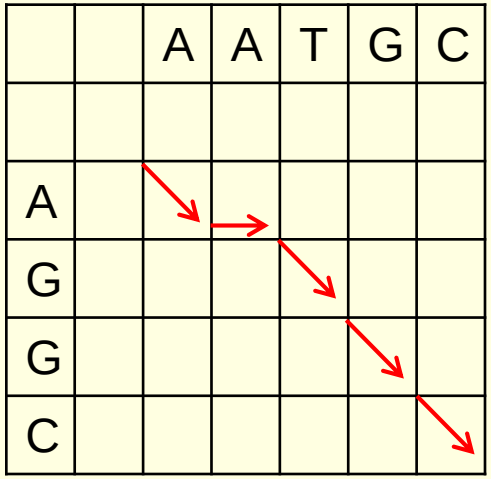
		A	A	T	G	C
		→				
A		↘	↘			
G			↘	→		
G					↘	
C						↘

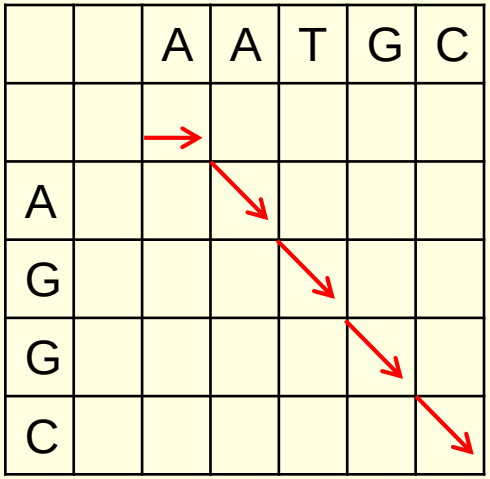
Path Matrix

Sequence Comparison

Dynamic Programming Alignment





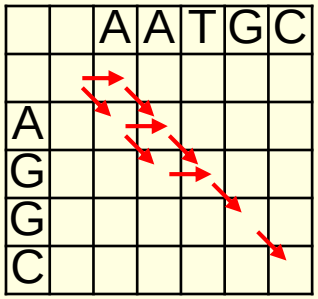


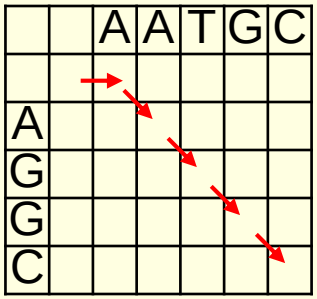
3 equivalent alignments, EVERY ONE IS OPTIMAL

```
AG.GC        A.GGC        .AGGC
AATGC        AATGC        AATGC
```

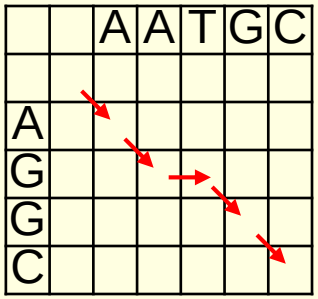
Sequence Comparison

Dynamic Programming Alignment

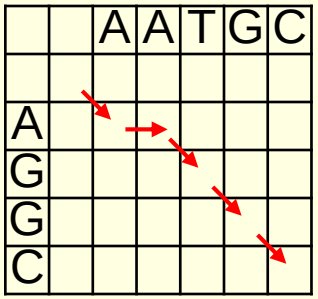





```
.AGGC
AATGC
```




```
AG.GC
AATGC
```

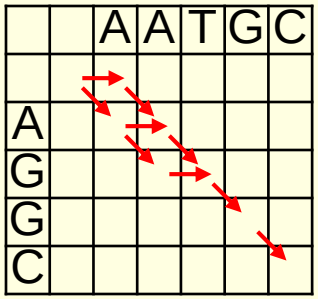



```
A.GGC
AATGC
```

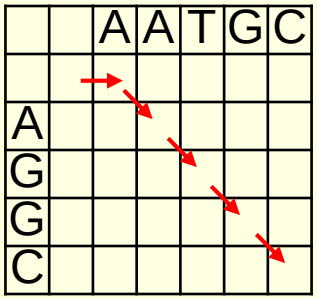
- 3 equivalent alignments, each one is optimal
- Most alignment programs will arbitrarily show you only one
- Most alignment programs will not tell your there are equally good alignments
- Dodgy spots are often at the edge of insertions/deletions (indels)

Sequence Comparison

Dynamic Programming Alignment

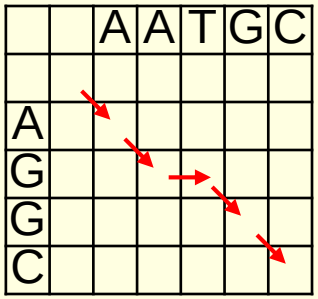


Trace upper path

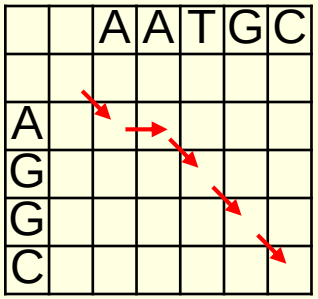


.AGGC
AATGC

Trace lower path

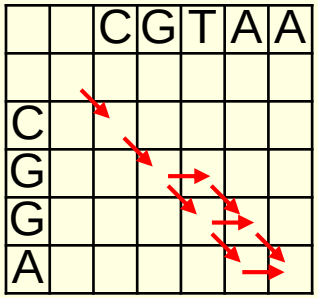


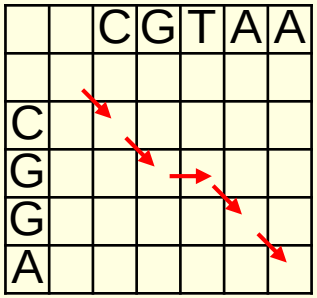
AG.GC
AATGC



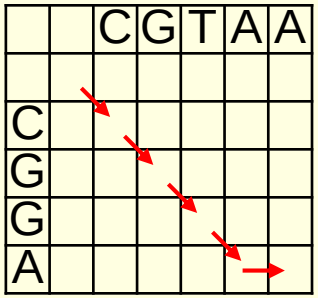
A.GGC
AATGC

- A trick: reverse the sequences and align. Due to the arbitrariness of which alignment is shown, the alignment will often be one of the equivalent ones. If they are different you know there are **AT LEAST** two equivalent alignments

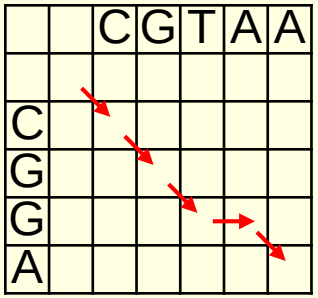




CG.GA
CGTAA



CGGA.
CGTAA



CGG.A
CGTAA

Sequence Comparison

Global Alignments

- ***AKA Needleman-Wunsch alignments (after the authors of the original paper)***
- ***Global alignments use all of the characters in both sequences***
- ***Unaligned characters (gaps) receive negative scores***
- ***Originally used linear gap-length dependent gap penalty (linear gap penalty)***
  - *Gap penalty = g = penalty x gap_length*

Sequence Comparison/Global Alignment

Negative scores at the edges reflect penalties for unaligned letters are the ends of the sequences

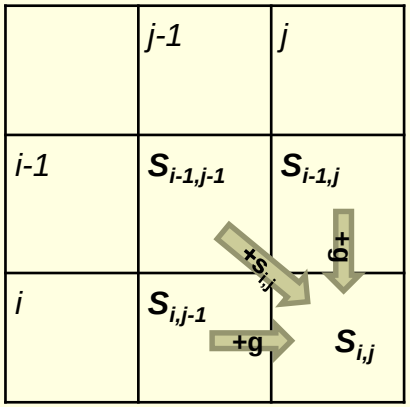
```
k u     m q u a t
. .     q u a n t i t y

. . .   k u m q u a t
q u a   n t i t y
```

		k	u	m	q	u	a	t	
	0	-1	-2	-3	-4	-5	-6	-7	
q	-1								
u	-2								
a	-3								
n	-4								
t	-5								
i	-6								
t	-7								
y	-8								

Score

Match (identity)	=	+2
Vowel/vowel	=	+1
Consonant/consonant	=	+1
Mismatch	=	-1
Gap	=	-1 / position

Sequence Comparison/Global Alignment



		k	u	m	q	u	a	t	
	0	-1	-2	-3	-4	-5	-6	-7	
q	-1								
u	-2								
a	-3								
n	-4								
t	-5								
i	-6								
t	-7								
y	-8								

$$S_{i,j} = \max \begin{Bmatrix} S_{i-1,j-1} + s_{i,j} \\ S_{i-1,j} + g \\ S_{i,j-1} + g \end{Bmatrix}$$

Eq 5.17,pg 131

Score

Match (identity)	=	+2
Vowel/vowel	=	+1
Consonant/consonant	=	+1
Mismatch	=	-1
Gap	=	-1 / position

Sequence Comparison/Global Alignment

$$S_{i,j} = \max \begin{cases} S_{i-1,j-1} + s_{i,j} \\ S_{i-1,j} + g \\ S_{i,j-1} + g \end{cases}$$

		k	u	m	q	u	a	t	
	0	-1	-2	-3	-4	-5	-6	-7	
q	-1	1	0	-1	-1	-2	-3	-4	
u	-2								
a	-3								
n	-4								
t	-5								
i	-6								
t	-7								
y	-8								

$S_{i-1,j-1} + s_{i,j} = 0 + 1 = 1$
$S_{i-1,j} + g = -1 - 1 = -2$
$S_{i,j-1} + g = -1 - 1 = -2$

Score
- Match (identity) = +2
- Vowel/vowel = +1
- Consonant/consonant = +1
- Mismatch = -1
- Gap = -1 / position

Sequence Comparison/Global Alignment

		k	u	m	q	u	a	t	
	0	-1	-2	-3	-4	-5	-6	-7	
q	-1	1	0	-1	-1	-2	3	-4	
u	-2	0	3	2	1	1	0	-1	
a	-3	-1	2	2	1	2	3	2	
n	-4	-2	1	3	3	2	2	4	
t	-5	-3	0	2	4	3	2	4	
i	-6	-4	-1	1	3	5	4	3	
t	-7	-5	-2	0	2	4	4	6	
y	-8	-6	-3	-1	1	3	5	5	

Score

Match (identity)	=	+2
Vowel/vowel	=	+1
Consonant/consonant	=	+1
Mismatch	=	-1
Gap	=	-1 / position

Sequence Comparison/Global Alignment

```
ku.mquat.
+|-+++-|-      =5
quanti.ty
```

```
ku.mquat.
+|-++-+|-      =5
quant.ity
```

		k	u	m	q	u	a	t	
	0	-1	-2	-3	-4	-5	-6	-7	
q	-1	1	0	-1	-1	-2	-3	-4	
u	-2	0	3	2	1	1	0	-1	
a	-3	-1	2	2	1	2	3	2	
n	-4	-2	1	3	3	2	2	4	
t	-5	-3	0	2	4	3	2	4	
i	-6	-4	-1	1	3	5	4	3	
t	-7	-5	-2	0	2	4	4	6	
y	-8	-6	-3	-1	1	3	5	5	

Score

Match	=	+2
Vowel/vowel	=	+1
Consonant/consonant	=	+1
Mismatch	=	-1
Gap	=	-1 / position

Sequence Comparison

Dynamic Programming Alignment

- ***Semi-global alignments***
  - The gaps at the end are “special” in some senses
    - Many proteins differ at the ends with little effect on structure
    - One sequence may be shorter than the other
    - Alignment may only include some overlapping portion
  - Using a gap cost=0 for end gaps was an early approximation to local alignment. The text calls this a semi-global alignment but this term is not widely used.
  - Using end gap cost=0 causes artifacts in alignments

```
M E Q N S L L V          instead of          M E Q N S L L V
. . . M E I L M                              M E . . . I L M
```

Sequence Comparison

Local alignments

- ***AKA Smith –Waterman after authors of original paper***
  - Must use positive and negative scores
    - Average score must be negative
  - Values are truncated at zero
  - Maximum can be anywhere in the score matrix,
    - Not just lower left corner

Sequence Comparison/Local Alignment

Cutting off scores at zero means that every similar diagonal starts at the same place; local regions are not penalized for previous gaps or mismatches. Note that this scoring system technically does not meet the requirements for a local alignment (average = 1.04)

		k	u	m	q	u	a	t	
	0	-1	-2	-3	-4	-5	-6	-7	
q	-1	1	0	-1	-1	-2	-3	-4	
u	-2	0	3	2	1	1	0	-1	
a	-3	-1	2	2	1	2	3	2	
n	-4	-2	1	3	3	2	2	4	
t	-5	-3	0	2	4	3	2	4	
i	-6	-4	-1	1	3	5	4	3	
t	-7	-5	-2	0	2	4	4	6	
y	-8	-6	-3	-1	1	3	5	5	

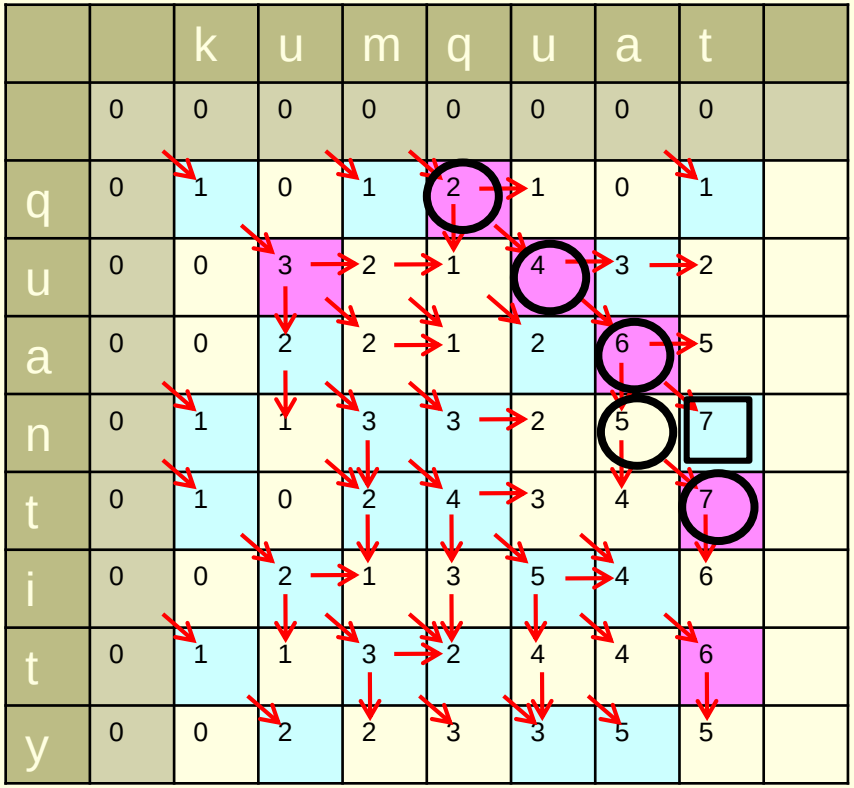
		k	u	m	q	u	a	t	
	0	0	0	0	0	0	0	0	
q	0	1	0	1	2	1	0	1	
u	0	0	3	2	1	4	3	2	
a	0	0	2	2	1	2	6	5	
n	0	1	1	3	3	2	5	7	
t	0	1	0	2	4	3	4	7	
i	0	0	2	1	3	5	4	6	
t	0	1	1	3	2	4	4	6	
y	0	0	2	2	3	3	5	5	

Score

Match (identity)	=	+2
Vowel/vowel	=	+1
Consonant/consonant	=	+1
Mismatch	=	-1
Gap	=	-1 / position

Sequence Comparison/Local Alignment

```
qua.t
|||-| = 7
quant

quat
|||+ = 7
quan
```



		k	u	m	q	u	a	t	
	0	0	0	0	0	0	0	0	
q	0	1	0	1	2	1	0	1	
u	0	0	3	2	1	4	3	2	
a	0	0	2	2	1	2	6	5	
n	0	1	1	3	3	2	5	7	
t	0	1	0	2	4	3	4	7	
i	0	0	2	1	3	5	4	6	
t	0	1	1	3	2	4	4	6	
y	0	0	2	2	3	3	5	5	

Score

Match	=	+2
Vowel/vowel	=	+1
Consonant/consonant	=	+1
Mismatch	=	-1
Gap	=	-1 / position

Sequence Comparison/Local Alignment

```
qua.t
|||-| = 7
quant
```

Local alignment with a scoring system with average < 0. Is it more reasonable to give a higher score for matching vowels?

There are fewer vowels than consonants. If there is no relationship between the sequences, we expect to see consonant matching with consonant more often than vowel with vowel. Vowel with vowel is more surprising, a better indicator of a true relationship.

		k	u	m	q	u	a	t	
	0	0	0	0	0	0	0	0	
q	0	1	0	1	10	8	6	4	
u	0	0	11	9	8	20	18	16	
a	0	0	9	7	6	18	30	28	
n	0	1	7	10	8	16	28	31	
t	0	1	5	8	11	14	26	38	
i	0	0	6	6	9	16	24	36	
t	0	1	4	7	7	14	22	34	
y	0	0	6	5	5	12	20	32	

Score

Match	=	+10
Vowel/vowel	=	+5
Consonant/consonant	=	+1
Mismatch	=	-5
Gap	=	-2 / position

Sequence Comparison/Local Alignment

Because the mathematically optimal alignment may not be the “correct” biological one, we could ask whether we can also see good but suboptimal alignments?

What about the alternative alignment

```
ku.mquat
+|-+++-|
quanti.t

k   u   .   m   q   u   a   t
1   +10 -2  +1  +1  +5  -2  +10  = 24
q   u   a   n   t   i   .   t
```

		k	u	m	q	u	a	t	
	0	0	0	0	0	0	0	0	
q	0	1	0	1	10	8	6	4	
u	0	0	11	9	8	20	18	16	
a	0	0	9	7	6	18	30	28	
n	0	1	7	10	8	16	28	31	
t	0	1	5	8	11	14	26	38	
i	0	0	6	6	9	16	24	36	
t	0	1	4	7	7	14	22	34	
y	0	0	6	5	5	12	20	32	

Score

Match	=	+10
Vowel/vowel	=	+5
Consonant/consonant	=	+1
Mismatch	=	-5
Gap	=	-2 / position

Sequence Comparison/Local Alignment

The alternative (suboptimal) alignment is “blotted out” by the optimal alignment. We can recover the alternative alignment by removing the scores for all pairs found in the optimal alignment. This gives us a *non-intersecting alignment*.

We have to recalculate the shaded area.

		k	u	m	q	u	a	t	
	0	0	0	0	0	0	0	0	
q	0	1	0	1	10	8	6	4	
u	0	0	11	9	8	20	18	16	
a	0	0	9	7	6	18	30	28	
n	0	1	7	10	8	16	28	31	
t	0	1	5	8	11	14	26	38	
i	0	0	6	6	9	16	24	36	
t	0	1	4	7	7	14	22	34	
y	0	0	6	5	5	12	20	32	

Score

Match	=	+10
Vowel/vowel	=	+5
Consonant/consonant	=	+1
Mismatch	=	-5
Gap	=	-2 / position

Sequence Comparison/Local Alignment

Non-intersecting alignment has expected score = 24

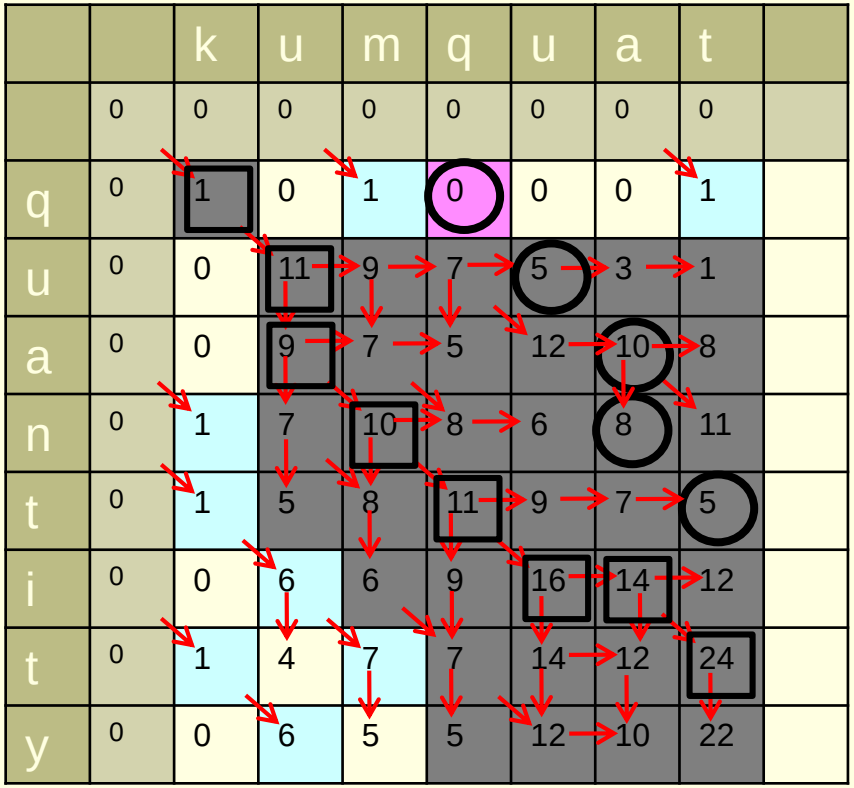
		k	u	m	q	u	a	t	
	0	0	0	0	0	0	0	0	
q	0	1	0	1	0	0	0	1	
u	0	0	11	9	7	5	3	1	
a	0	0	9	7	5	12	10	8	
n	0	1	7	10	8	6	8	11	
t	0	1	5	8	11	9	7	5	
i	0	0	6	6	9	16	14	12	
t	0	1	4	7	7	14	12	24	
y	0	0	6	5	5	12	10	22	

Score

Match	=	+10
Vowel/vowel	=	+5
Consonant/consonant	=	+1
Mismatch	=	-5
Gap	=	-2 / position

Sequence Comparison/Local Alignment

We can repeat this, eliminating the second alignment as well. The shaded area must be recalculated.



Score

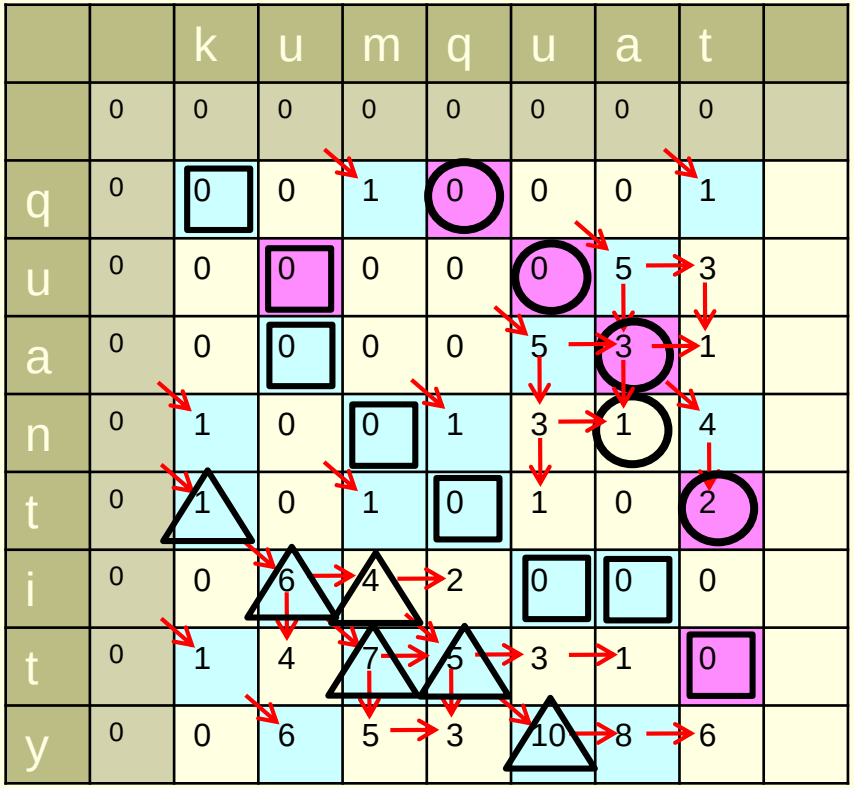
Match	=	+10
Vowel/vowel	=	+5
Consonant/consonant	=	+1
Mismatch	=	-5
Gap	=	-2 / position

Sequence Comparison/Local Alignment

Third non-intersecting alignment

```
Kumqu
+++-+
tit.y

Both score 10
```

```
kumqu
++-++
ti.ty
```



Score

Match	=	+10
Vowel/vowel	=	+5
Consonant/consonant	=	+1
Mismatch	=	-5
Gap	=	-2 / position

Week 2
26 – 30 August

- ***Monday - Alignments/Dynamic Programming pgs 127 - 139***
- ***Wednesday - Alignments /Scoring Systems***
  - Ch 4.3,5.1
- ***Friday – Alignments /Scoring Systems***
  - Be sure you have read Ch 5.1 for friday

Sequence Comparison

Dynamic Programming Alignment

- **If the alignment is short, it may be entirely missed in a global alignment!**

```
                .         .         .         .         .
151 VEKGGKHKTGPNLHGLFGRKTGQAPGYSYTAANKNKGIIWGEDTLMEYLE 200
                          ... .   |   |    . :   | ||
  1 .....................VLSPADKTNVKAAWGKVGAHAGEYGAEALE 30
          .         .         .         .         .
201 NPKKYIPGTKMIFVGIKKKEERADLIAYLKKATNE.............. 235
        .| ||  |  .   .  |.. .  || ..
 31 RMFLSFPTTKTYFPHFDLSHGSAQVKGHGKKVADALTNAVAHVDDMPNAL 80
```

Global using PAM250 scoring matrix
Gap opening = -12
Gap extension = -4

```
 52 LSDGEWQLVLNVWGKVEADIPGHGQEVL 79
    || .:   |   |||| |.    | | |
  2 LSPADKTNVKAAWGKVGAHAGEYGAEAL 29
```

Local using PAM250 scoring matrix
Gap opening = -12
Gap extension = -4

Sequence Comparison

Global vs Local Alignments

Global	Local
• Uses all positions of both sequences • Any scoring system • Trace from lower right corner	• Best region of alignment • Scoring system must have average less than zero • Negative scores truncated at zero • Trace from maximum in whole matrix

- ***Use global if***
  - You expect and want the entire sequence to match, *e.g.* two close homologs
- ***Use local***
  - You expect only part of the sequences to match, or you don't know
  - Sequences are very different in length
  - Alignment may be only some overlapping section

Sequence Comparison

Global vs Local Alignments

Global	Local
• Uses all positions of both sequences • Any scoring system • Trace from lower right corner	• Best region of alignment • Scoring system must have average less than zero • Negative scores truncated at zero • Trace from maximum in whole matrix

- ***Use global if***
  - You expect and want the entire sequence to match, *e.g.* two close homologs
- ***Use local***
  - You expect only part of the sequences to match, or you don't know
  - Sequences are very different in length
  - Alignment may be only some overlapping section

Sequence Comparison

Dynamic Programming Alignment Review

- ***Dynamic programming alignment is mathematically rigorous***
  - considers all possible alignments with all possible gaps in all possible positions
- ***Two steps***
  - Build score matrix - each cell represents the score for the best alignment that ends in that position
  - Follow “best previous” pointers back to get alignment (traceback)

Points to remember

- ***There may be other equally good alignments***
- ***Highly dependent on the scoring system***
- ***Gap penalties have a large effect on alignments***
  - Do not rely on a single setting
- ***Are you using a global or local method ?***
- ***Are end gaps weighted or unweighted (semi-global)?***
- ***How does your score compare to unrelated sequences?***

Sequence Comparison

Getting quantitative

- ***How similar does a pair of sequences have to be before you can decide they are homologous?***

- ***Problems/Issues***
  - Long sequences generally get higher scores than short sequences
  - GC or AT rich sequences get higher scores when compared to GC or AT rich sequences, respectively
  - Scores depend on match/mismatch scores and gap penalties

Sequence Comparison

Getting quantitative

- ***If the score for an alignment is no higher than we would see for unrelated sequences, we are not impressed***
  - The sequences are unlikely to be homologous
- ***If the score is so high that it is very unlikely to occur between unrelated sequence…***
  - The sequences are unlikely to be unrelated, or in other words, likely to be homologous

Sequence Comparison

Scoring - Probabilistic interpretation

```
Seq 1  ATATGCA
Seq 2  AAATCCA
```

- ***Two possibilities:***
  - The sequences are homologous, related by descent from a common ancestor with mutations
  - The sequences are unrelated – essentially random (*Null hypothesis*)
- ***If the match we see is very unlikely (= very low probability) we can reject the null hypothesis and infer they are homologous.***
- ***Simplistic view: the probability of two random sequences matching is the product of the probability of drawing the pairs of residues at random***
- ***P(Seq1 homologous to Seq2) = Π P(pair$_i$) for i=1,n***

Sequence Comparison

Probabilistic interpretation

```
Sequence 1   ATATGCA
Sequence 2   AAATCCA
```

- *P(Seq1 homologous to Seq2) = Π P(pair$_i$) for i=1,n*
- *Probability of random match = 4/16 = 0.25*
- *Probability of mismatch = 12/16 = 0.75*
- *Probability of match with 5 matches + 2 mismatches $(0.25)^5 \times (0.75)^2 = 0.00055$*
- *Log(P(match)) = Σ log(P(pair))*

- *Score for alignment = Σs_{ij}*
- *Intuitively, scores have some relationship to log probabilities where the probabilities tell you something about how likely you are to align the letters by random chance*

Sequence Comparison

Alignment significance

- ***If a sequences are significantly more similar than random sequences, they must be homologous sequences***
- ***Significance can be evaluated using a random sequence model and a Monte Carlo procedure***
- ***Compares result to randomized result, similarly to results generated by a roulette wheel at Monte Carlo***
- ***Typical procedure for alignments***
  - Randomize sequence A
  - Align to sequence B
  - Repeat many times (hundreds - thousands)
  - Use average as expected score to calculate Z scores

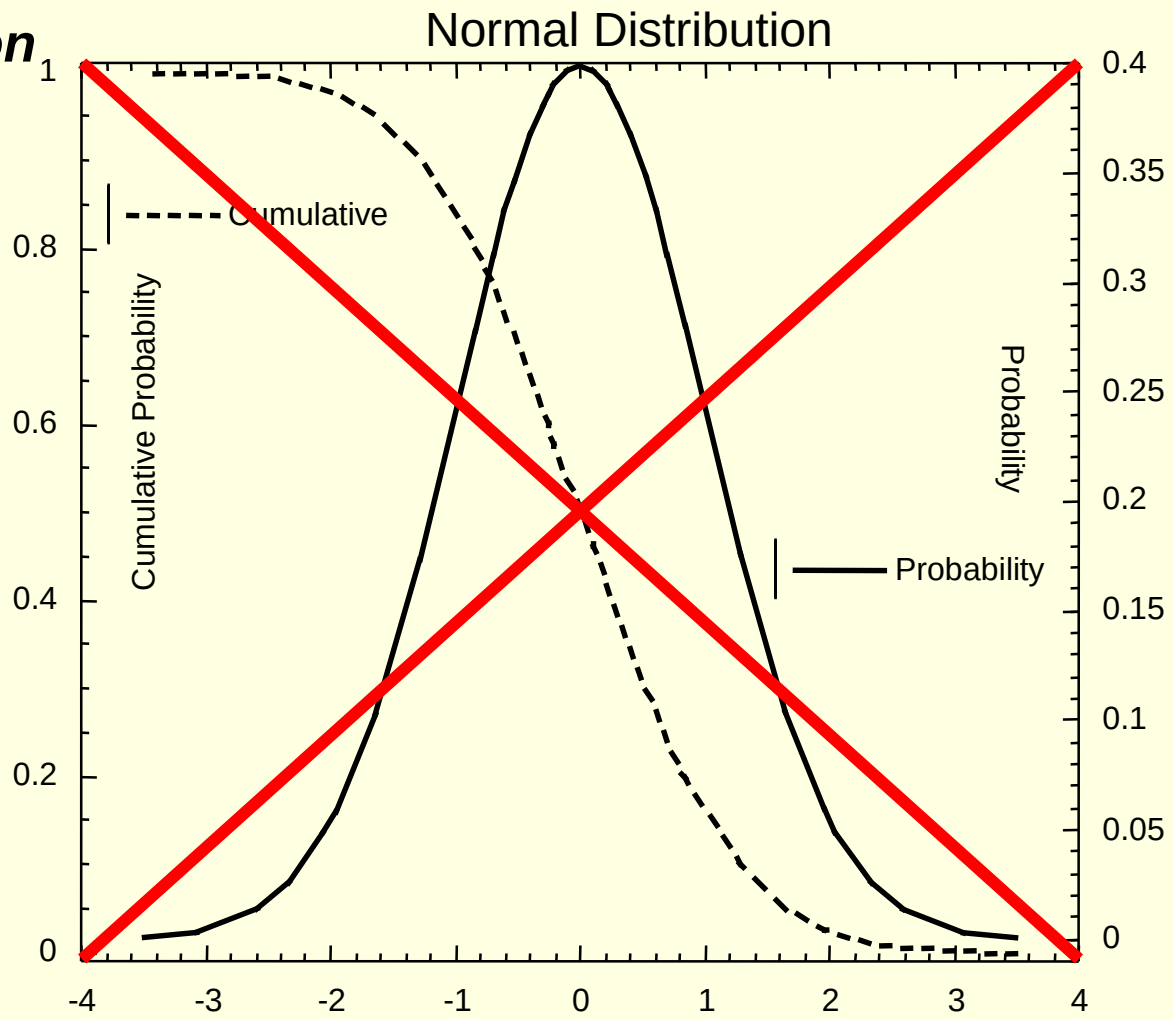
Sequence Comparison

Alignment significance

- ***Alignments are often evaluated using a random sequence model and a Monte Carlo procedure***
- ***If a sequences are significantly more similar than random sequences, they must be homologous sequences***
- ***A common statistic is the Z score (standardized score, standard normal deviate)***
  - Z = (Obs_score - Exp_score) / Std_deviation
  - Expected score depends on scoring and penalties
- ***Old Rule:***
  - ~~Z < 3~~ No evidence of homology
  - ~~3 < Z < 6~~ Homology possible
  - ~~6 < Z~~ Strong evidence of homology, (Z > 8) better

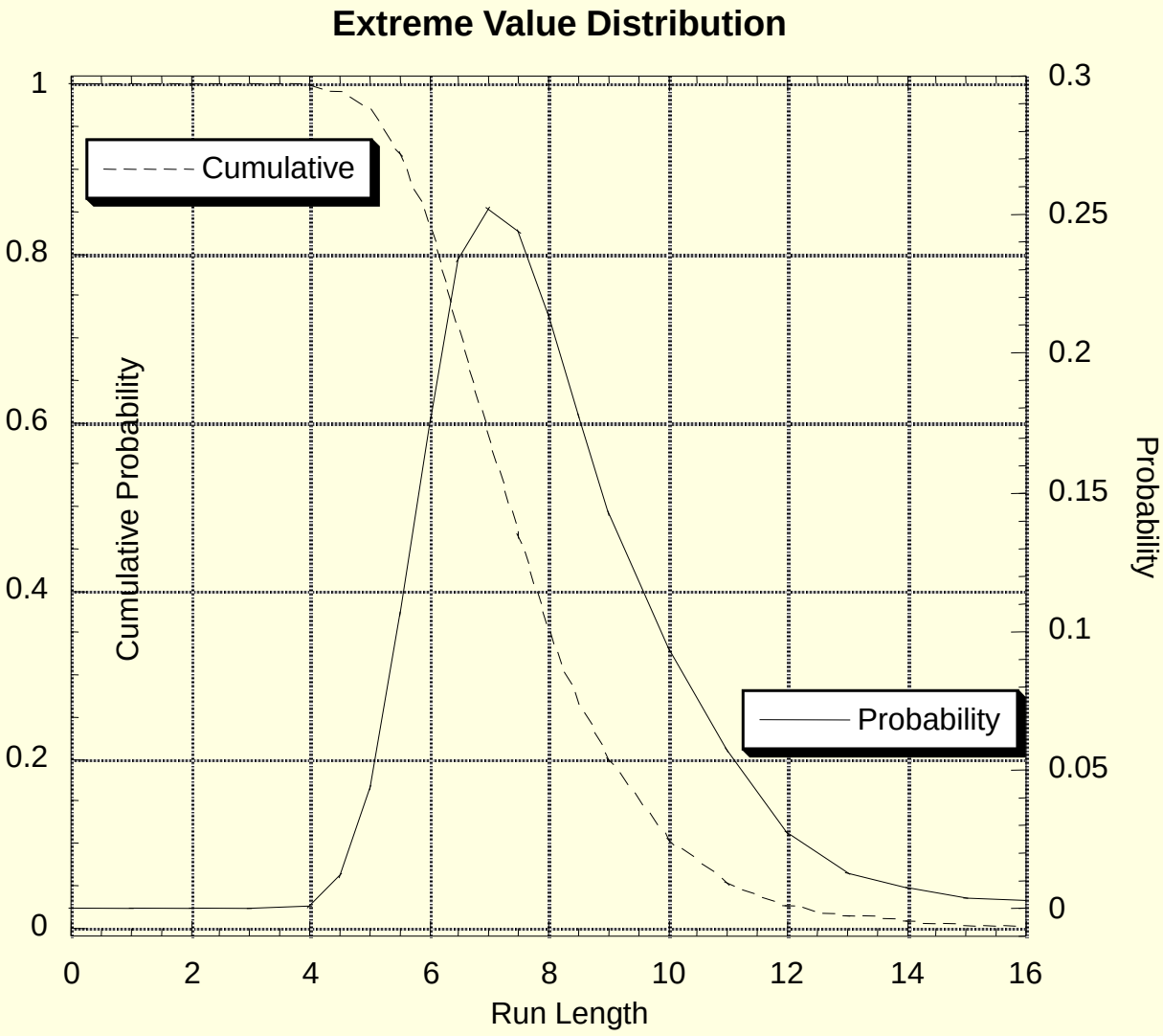
Sequence Comparison

- ***Z score assumes that the distribution is a normal (gaussian) distribution***



Sequence Comparison

- ***Sequence alignments and other maximizations do not follow normal distributions***



Sequence Comparison

Extreme Value Distributions

- ***Whenever you are looking at a distribution of maxima***
  - longest run of heads in coin toss
  - maximum scores for each sequence in database
- ***Z score can't be directly converted to probability because it not a "Normal" or "Gaussian" distribution***
  - *e.g.,* Z=3 has a normal P-value = 0.0013 but an extreme value distribution P-value ~ 0.12
  - about 100-fold error (error gets worse for smaller P-values)

Sequence Comparison

- ***Updated Monte Carlo procedure for dynamic programming alignments***
- ***Karlin-Altschul statistics predict significance of extreme value distributions***

  $n \sim KNe^{-\lambda S}$ where S is the alignment score and n is the number of alignments with equal or higher scores

  taking the log of this equation

  $\ln(n/N) = \ln K - \lambda S$ this is a linear equation

  $y = b + mx$
- ***Plot the log of the "observed P-value" vs score for randomized alignments of the same length and composition and determine P from the linear plot***
  - Randomize sequences and align using same parameters a large number of times, e.g. >10,000. Rank results by score. Observed P-value is rank divided by number of samples

Sequence Comparison

Alignment scores

- *Calculate using dynamic programming alignment*
- *Evaluate empirically*
- *Interpret in terms of probability vs probability of unrelated sequences*

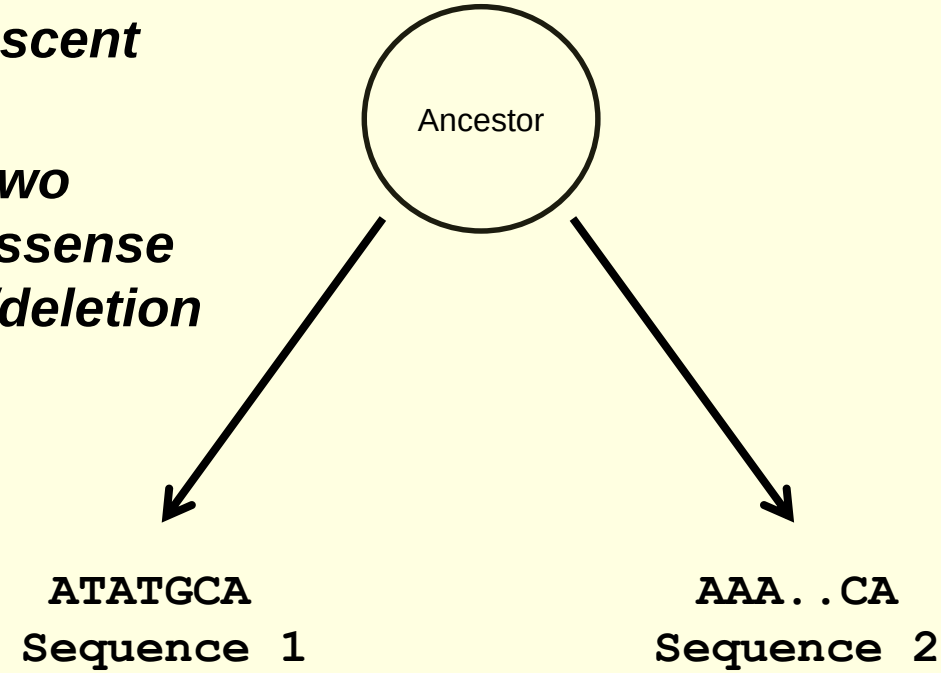
- *Intuitively scores are related to these probabilities*

Sequence Comparison

Alignment Model

```
Sequence 1    ATATGCA
Sequence 2    AAA..CA
```

- *Sequence 1 and Sequence 2 are related to each other by descent from a common ancestor*
- *Along the way there were two mutational events: one missense change, and one insertion/deletion (indel)*



Sequence Comparison

Measuring the difference between sequences

```
ACGGTTAGCAAA      ACGGTTAGCAAA      ACGGTTAGCAAA
|||||||  ||||     ||||  ||  ||||     ||   |||  ||||
ACGGTTACCAAA      ACGGATACCAAA      ACCCTTACCAAA
```

1	2	3	Distance	Match Score
11	10	9	Similarity	

```
ACGGTTAGCAAA      ACGGTTAGCAAA      ACGGTTAGCAAA      ACGGTTAGCAAA
 |          ||     |   |    |||       |    |||          |||  ||||
CCTTACCAAAAC      CCTTACCAAAAC      CCTTACCAAAAC      CCTTACCAAAAC
```

Sequences may have to be offset to optimize score

Sequence Comparison

Scores

- *Scores for match and mismatch correspond to scores for bases/residues that are conserved or changed by missense mutations (assuming homology)*
- *What do scores for gaps represent?*
- *How big should (negative) scores for gaps be relative to missense changes?*
  - *Missense mutations are much more common than insertions and deletions*
  - *Gaps should therefore have bigger negative scores than mismatches*

Sequence Comparison

Measuring the difference between sequences

- *What if sequences need gaps, what should the score be?*

```
AGTTACGGCAAA      AGTTACGGCAAA      AGTTACGGCAAA
|||||     |             |||||       |||||   |||||
AGTTAGCAAA        AGTTAGCAAACC      AGTTA..GCAAACC
```

```
ATCTAGCAG.T.C.A
  ||  |  ||  |  |  |
  CT.G.AGCTCCCA
```

- Allowing gaps makes it easier to get a high score.
- Intuitively, there should be some negative score for gaps.
- Otherwise any pair of random sequences can get a high score.

Sequence Comparison

Measuring the difference between sequences

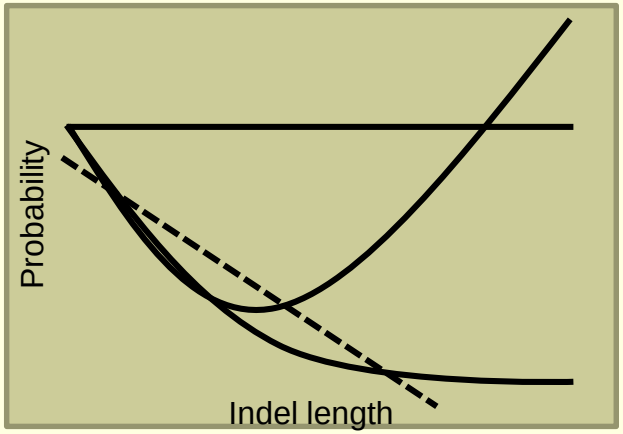
- *Sequences alignments use a scoring function based on the number of matches and mismatches, and a function based on the number of gaps*

$$Match = N_{match} - N_{mismatch} - f(gap)$$

- *The score than unrelated sequences might get (on average) also matters*

Sequence comparison

Gap penalties



- ***How does the probability of indels vary with length?***
- ***What do we think?***
  - Long indels are common, although less common than short indels
  - Over some length, length makes little difference, probably because long indels and short indels are caused by different biological processes.
- ***Alignments use a linear model for gap penalties called an Affine Gap Cost***

$$Penalty = G_{open} + Length * G_{extend}$$

Sequence Comparison

Gap Penalties

- ***Why use affine gap cost?***
  - Experience shows that neither a length independent or length dependent penalty, by itself, works well, but that affine gap penalties do
  - More complicated gap functions (such as concave upward functions) are more difficult to implement and increase the computational cost of alignments from $O(n^2)$ to $O(n^3)$

- ***Recommended values for alignments***
  - G_{open} (gap opening; GOP) penalty approx 3 x identities
  - G_{extend} (gap extension; GEP) about 5% or less of G_{open}

Sequence Comparison

Gap Penalties

- ***Why use affine gap cost?***
  - Experience shows that neither a length independent or length dependent penalty, by itself, works well, but that affine gap penalties do
  - More complicated gap functions (such as concave upward functions) are more difficult to implement and increase the computational cost of alignments from $O(n^2)$ to $O(n^3)$

- ***Recommended values for alignments***
  - G_{open} (gap opening; GOP) penalty approx 3 x identities
  - G_{extend} (gap extension; GEP) about 5% or less of G_{open}

Week 2
26– 30 August

- ***Monday - Dotplots***
- ***Wednesday - Alignments/Dynamic Programming***
- ***Friday – Alignments /Scoring Systems***

Week 3

- ***Reading***
  - Ch 4.6-4.7, 5.3-5.4
- ***Monday 9/3 - labor day holiday***
- ***Wednesday - Quiz***
- ***Friday 9/7 - finish alignments and practical examples***

Scoring Systems

- ***Simple matching of identical characters (identities) is usually sufficient for DNA.***
  - Scoring systems including transition/transversion are possible and sometimes used (discussed on pg 125 in text)
- ***Proteins residues have complex patterns of similarity based on the chemistry of the amino acid side chains.***
- ***Protein comparisons generally use a 20 x 20 comparison table to describe the similarity of the amino acid residues.***
- ***Two major scoring systems for proteins:***
  - PAM or Dayhoff (AKA MDM78, PAM-250, PAM-125)
  - BLOSUM series

Scoring Systems

Simple Systems – scoring based on a property

- ***The simplest scoring systems score some kind of property, hydrophobicity is a good example.***

R	***-4.5***	***S***	***-0.8***
K	***-3.9***	***T***	***-0.7***
D	***-3.5***	***G***	***-0.4***
Q	***-3.5***	***A***	***1.8***
N	***-3.5***	***M***	***1.9***
E	***-3.5***	***C***	***2.5***
H	***-3.2***	***F***	***2.8***
P	***-1.6***	***L***	***3.8***
Y	***-1.3***	***V***	***4.2***
W	***-0.9***	***I***	***4.5***

If we want to make an alignment that maximizes similarity in hydrophobicity we could use a scoring system based on the difference in hydrophobicity, for example (9.0 is maximum difference)

$$s_{ij} = 9.0 - (H_i - H_j)$$

- ***Intuitively, comparisons that are very large in our group of interest, and rare in other sequences are most informative – that is, those comparisons are the most surprising***
- ***To tell how surprising a score is, you must have an idea of what kinds of scores you expect to see in sequences. If you are looking for transmembrane sequences, you must know what scores are typical of non-transmembrane sequences***

Scoring Systems

Simple systems – classification into two groups

- *Is a sequence a part of a transmembrane region?*
- *We can use the frequencies that we observe a specific sequence in transmembrane regions and non-transmembrane regions to make a classifier (or discriminator)*
- *For instance, consider the three residue sequence IVI and the sequence IAG. Say we count these up and find the following*

sequence	#tm	#non-tm
IAG	3/1000	4/1000
IVI	12/1000	1/1000

- *Clearly the IVI sequence makes a better discriminator.*

Scoring Systems

Simple Systems – classification into two groups

- *A simple way to capture our qualitative sense that IVI is more discriminating that IAG is to look at the ratio of the frequency of each sequence in the TM and non-TM classes:*

  $R_{IAG} = F_{IAG,TM} / F_{IAG,NTM} = 0.003/0.004 = 0.75$
  $R_{IVI} = F_{IVI,TM} / F_{IVI,NTM} = 0.012/0.001 = 12.0$

- *If you looked at another sequence, e.g., AST and found*

  $$R_{AST} = F_{AST,TM} / F_{AST,NTM} = 0.001/0.012 = 0.008$$

  we have the beginning of a system to discriminate TM and non-TM regions.

- *However, note that if you add up these scores, the scores favoring TM regions will dominate those favoring non-TM regions even though the degree of difference in occurrence is similar as for IVI and AST.*

- *This can be simply adjusted for by taking the log of the ratio, i.e.,*

  $\ln R_{IVI} = 2.48$ $\ln R_{AST} = -2.48$

- *This is a log-odds scoring system.*

Scoring Systems

Alignments

- *For alignments, we want a scoring system that discriminates between sequences that represent homology, and unrelated sequences. A log-odds scoring system makes sense*

- *For alignments we are interested in the frequencies of the amino acid pairs in alignments of homologous proteins vs what we expect to see in unrelated proteins*

```
52 LSDGEWQLVLNVWGKVEADIPGHGQEVL 79
   || .:   |   |||| |.    | | |
 2 LSPADKTNVKAAWGKVGAHAGEYGAEAL 29
```

That is L:L, S:S, D:P, G:A, … (20x20 possibilities)

- *We can count their frequencies is a set of classified examples of true homologous sequence pairs and unrelated sequence pairs to construct a log-odds scoring system*

Scoring Systems

Log-Odds Scoring

- ***Log-Odds scoring systems compare two models, a foreground model, Q, and a background model, P. The background model is often a random sequence model.***

- ***The score for comparing two residues, s_{ij}, is the log of the ratio of the foreground and background probabilities, i.e., how many times more likely than chance you are to see the residues in the matched in the foreground model.***

$$s_{ij} = \ln(q_{ij} / p_{ij})$$

- ***Where q_{ij} and p_{ij} are the scores for comparing residues i and j in the foreground and background models, respectively***

Scoring Systems

Log-odds scoring

- *A log-odds scoring system evaluates the relative probabilities of a match representing true homology versus the chance that a match occurs at random, i.e. the relative probability of two models*

$$s_{ij} = \ln(q_{ij} / p_i \, p_j)$$

- *Normally, one multiplies probabilities - since these are log probabilities you get the overall (relative) probability by adding them up*

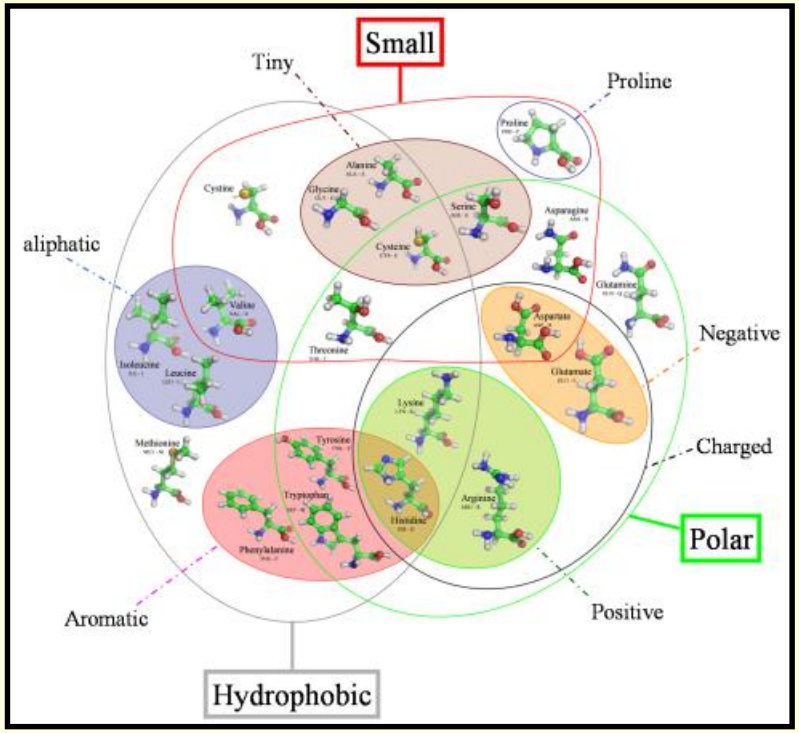
- *When added up over a matching segment, as an alignment does, you get the probability that the segment represents homology relative to the probability that it represents a random match, i.e., how much more likely than chance is it that the matching segment represents homology*

Scoring Systems

Why use a non-identity matching system?

- *Identity is fine for DNA, but does not recognize chemical/structural similarity of amino acid residues*

- *Does not take evolutionary distance into account*



Scoring Systems

PAM vs BLOSUM

- ***Empirical scoring systems work better than all know theoretical scoring systems (for proteins). There are two popular <u>empirical</u> scoring systems for protein sequences***

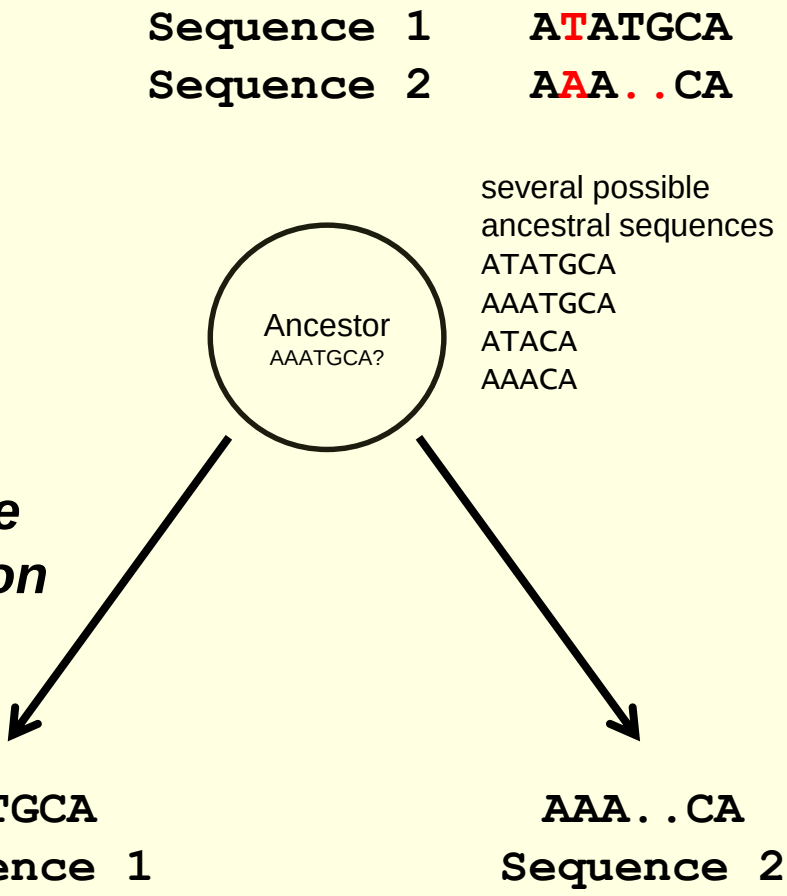
- ***PAM (Percent Accepted Mutation)***
  - Based on explicit evolutionary model
  - Represents a specific evolutionary distance
  - Ranges from identical to completely random

- ***BLOSUM (BLOcks SUbstitution Matrix)***
  - Based on empirical frequencies
  - Always a blend of distances as seen in the database and PROSITE
  - Narrower range than PAM matrix

Sequence Comparison

Alignment Model

- ***Empirical scoring systems are based on a simple model of sequence evolution***
- ***Sequence 1 and Sequence 2 are related to each other by descent from a common ancestor***
- ***Along the way there were two mutational events: one missense change, and one insertion/deletion***



Scoring Systems

Effect of multiple mutations

- *Evolutionary distance is estimated from number of differences*
- *Multiple mutations make counting mutations difficult*
- *Below are sequences generated by random mutations with 1% probability/gen.*

```
 10 AAGCGCCGTCACTCGCGTATAAGCCCCGCAATACGACCCGCTTTGCACGCCTCGGGCACGACGGGGCAACGCGAGCGCCAGCCGCAGGAGGTTGCGTACG
 20 AAACGCCCTTACTCGCGTATAAGCCCCGCAATAGGACCCCCTTTGCAGGCGTCGGGCACGACGGGGAAACGCGAGCGCCAGCCGCAGGAGGTTGCGTACG
 30 AGGCGCACTTACTCGCGTTTAAGCCCCGCACTAGGACCCGCTTTGGATGCGTCGGGCAGGACTGGGAAACGCGAGCACCAGCCGCACGAGGTTGCGTACG
 40 AGGCGCACGTACCCGCTTTTAAGCCCCGCACTCGGACCCGTTTTGGATGCGTCGGGCAGGACTGGGAAAAGCGAGCACCAGCGGCACGACGTCGCGTACG
 50 AGGCGCACGTACCAGGGTTTAAGCCCCGCACTCCGACCCGTTTTGGATTGGTCGGGCAGGCCTGGGAAAAGCGAGCACCGGGGACACGACGTCGCGTACG
 60 AGGCGCACGTACCAGGGTTTAAGCCCCCCACTCCTACCCGTTTTGGATTGTTCGGGGAGGCCTAGGAAAAGCGACAACCGGGGACCCGACGTCGCGTACG
 70 AGGAGCACGTACCAGGGTCAGAGTCCCCCACTCCTACCCGTTTTGGCTTGTTCGGGGAGGCCTAGGAAAAGCGAGAGCCGGGGACGCGACGTCGCGTATG
 80 AGGAGCACGTACCAGGGTCAGAGTCCCCCACTCCTGCCTGATTTGGCTTGTTGGGAGAGGCCTAGGAAAAGCGCGAGCCGGGGACGCGACGTCGCGTATG
 90 AGCAGCACGTACCAGGGTCAGATCCCCCCCCTCCGGCCAGATTTGGCTCGTTGGGGGAGGCCTAGGTAAAGCGCGATGCGTGGCCGCGACGTCGCGTATG
100 AGGAGCACGTACCAGGGTCGGATCCCCCCCCTCCGGCCCTCATTCTATCGTTGGCGGAGGTCTAGGTAAAGCCCGATGCCTGGCCGCGACGTCGCGTATG
110 AGGAGCACGTACCAGGCTCGGATCGCGCCCCGCAGGCCGTGATTCCATAGTTGGCGGAGGTCTAGGTAAAGCCCGATTCCTGGCCGCGACGTCGCGTAGA
120 AGGAGCACGTACCAGGCTCGGATCGCTCACCCCAGGCCGTGATTTCATAGATGGCGGAGGTCTAGGTCAAGCCCGATTCCTGGCCGTGACGTCGCGTAGA
130 CGGAGCACGTACCAGGCCCGGATCGGTCAGCCCAGGCCGTGATTACATACATGGGCGGAGGTCTGGGGCAAGGCCGATTACTGGCCGTGACGTCGCGTAGA
140 CGTCGCACGTACCAGGCCCTGATCGGCCAGCCCAGGCCCTGATTCCATACATCGCTGAGGTCTGGGGCAGGGCCGATTACAGGCCGTTACGTCGCGTAGA
150 CGTCGCAGGTACCAGGCCCTGTTCGGCCAGCCCAGTCCATCATTCCATACATCGGTTAGGCCTGGGGCAGGCCCGATCACAGGCCGTTCCGTCGCGTAGA
160 CGTCGCAGGTACCAGGCCCTGTTCGGCCAGCCCACTCCATCATACCATACATCCCTTAGGCCTGGGACCCGCCGGATCACATGCCGTTCAGTCGCGTATA
170 CGTCAGAGGTACCAGGCCATGTTCGGCCAGCCCATTCTCTCATACCATACATCCCTTAGGAGTGGGACCCGCCGTATCGAATGCCGGTCAGTTGCATATA
180 CCTCAGAGGTACCGGCTATGTTCCGCGAGCCCATTGTATCTAACCCTACATCCCTTAGGAGTGGGACCCGCGGTATCGAAGACCGGTCAGTTGCATCTG
190 CCTCAGAGGTAGCGGCTATGTTCCACGAGCCGAATGTATCTAACCCTACATCCCTTAGGGGTGGGACCCGCGGGATCGAATACCGGTCAGTTGCATCTG
200 CCTCAGAGGTAGCGGCTCTGTTCCACGAGCTTAATGTATCTGACCCTACATCCCTTAGGGGTGGGACCCGCTGGATCGAATACCGGTCAGTTGCCTCTC
210 CCTCAGAGGTAGCGGCTCTGTTACACGAGCTTTATGTATCTGACCCTACATCGCTTAGGGGTGGGACCAGCAGCATCGGATACCGGTCAGTTGCCTCTC
220 CCTCAAAGGTAGCGGCCCTGTTACACGCGCTTGATCTATCTGACCCTTCATCGCTTAGGTGTGGGACCAGCAGCAACGGATACCGGTCAGTTGCTTCCC
```

Single Mutation

Mutate, and mutate back

No change

Multiple mutations

Never mutated

Scoring Systems

PAM Matrices

- ***PAM model of evolution was originally proposed by Margaret Dayhoff and co-workers in the 60's***
- ***This work has proven to be very enduring, it is still difficult to do better than this group did in the sixties***
- ***Approach:***
  - Carefully examine the kinds of mutations that occur in closely related protein sequences, *i.e.*, at short evolutionary times
  - Extrapolate these differences to greater mutational distance/longer times

Scoring Systems

PAM Matrices

- ***PAM means "percent accepted mutations"***
  - accepted means fixed in the population and is therefore a more complex process than simply mutation
- ***PAM-1 therefore is a scoring system for sequences in which 1% of the residues have undergone mutation***
- ***PAM-250 represents 250% mutation, i.e., an average of 2.5 accepted mutation per residue - a very distant relationship***
- ***PAM tries to model what happens at long evolutionary distances based on a simple Markov model derived from closely related sequences.***

Scoring Systems

PAM Matrices

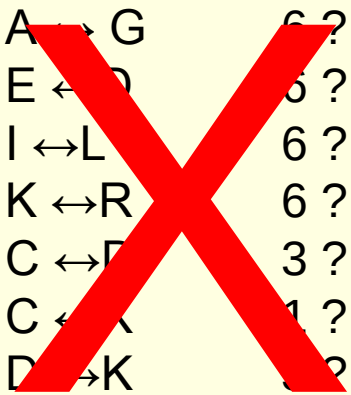
- ***Accepted point mutations - tabulate actual mutations by looking at proteins that are sufficiently closely related that there is no ambiguity in alignment***
  - Relying on actual observed mutations is why we call it empirical
  - Sequences no more than 15% different so that changes can be thought of as a single evolutionary step – no multiple mutations or back mutations
  - 1572 changes in 71 families
  - Consider a tree to correctly count changes

Scoring Systems

PAM Matrices

- *Counting mutations. How many changes in these sequences?*

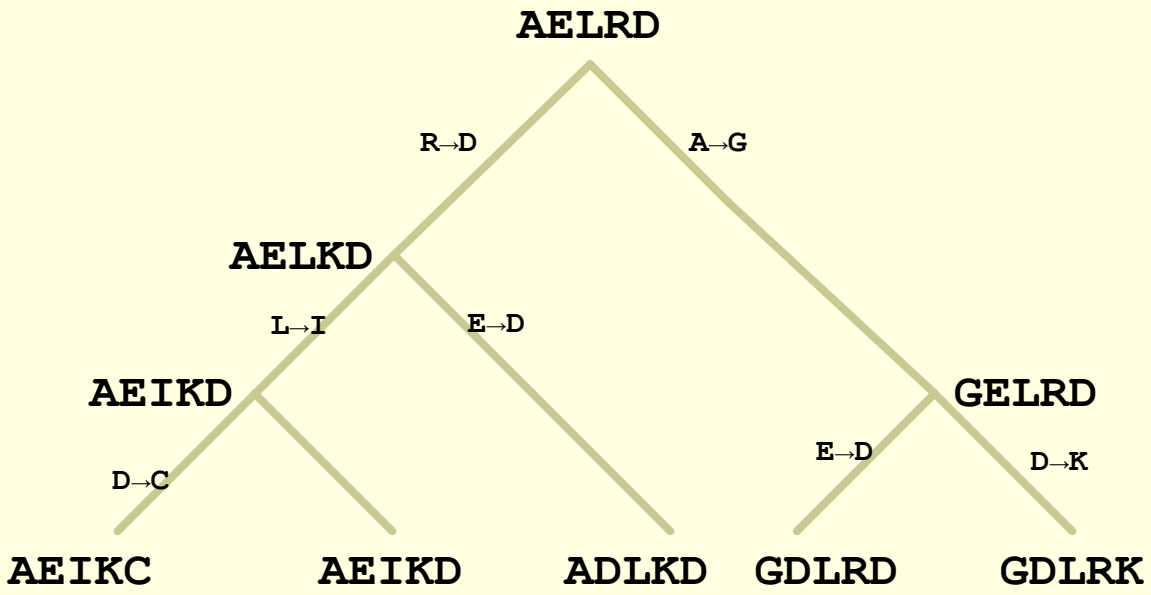
AEIKC
AEIKD
ADLKD
GDLRD
GDLRK



Scoring Systems

PAM Matrices

- ***Counting mutations (correctly)***



A→G	1
D →C	1
D →K	1
E →D	2
L →I	1
R →D	1

From Previous page

A ↔ G	6 ?
E ↔D	6 ?
I ↔L	6 ?
K ↔R	6 ?
C ↔D	3 ?
C ↔K	1 ?
D ↔K	3 ?

See also fig 5.1 in text

Scoring Systems

PAM Matrices

- **Mutation probability matrix - probability that residue in column j will be replaced by residue in row i after some amount of evolution**

  $M_{jj} = 1 - \lambda m_j$ $\quad\quad$ $M_{ij} = \lambda m_j A_{ij} / \Sigma A_{ij}$

  m_i = mutability of residue i (probability of mutating)
  A_{ij} = number of accepted point mutations
  λ = proportionality constant

- **λ is chosen to scale the number of mutations to a certain amount, Dayhoff chose 1% (or 1 PAM), see Eq. 5.6 in text**

  percent mutation = $100 \Sigma f_b (1 – \lambda m_b)$

Scoring Systems

PAM Matrices

- *1 PAM matrix can be extended to simulate longer distances by multiplying it by itself.*
- *Dayhoff recommended PAM 250*
- *More modern recommendations suggest PAM 125 is a better general choice*
- *Which is best depends on the sequences!*

Scoring Systems

PAM Matrices

- ***Mutation Probability Matrix***
  - Probability that the residue represented by the column will mutate into the residue represented by the row after a specified amount of mutation, for instance, 1PAM
  - Not symmetric. The probability of A→S ≠ S → A
- ***Relatedness odds matrix***
  - Log-odds form of the mutation probability matrix
    - Log-odds matrix compares what is expected of homologous sequences to what is expected of unrelated (random) sequences.
  - Remember that a log-odds matrix compares two models, in this case a model of relationship by homology (the mutation probability matrix) and relationship by chance

$$R_{ij} = log(M_{ij} / f_i)$$

  - Where f_i is the frequency of amino acid residue *i* at random

Scoring Systems

PAM Matrices

- *Original PAM 250 matrix (compare fig 5.3)*

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W
C Cys	12																			
S Ser	0	2																		
T Thr	-2	1	3																	
P Pro	-3	1	0	6																
A Ala	-2	1	1	1	2															
G Gly	-3	1	0	-1	1	5														
N Asn	-4	1	0	-1	0	0	2													
D Asp	-5	0	0	-1	0	1	2	4												
E Glu	-5	0	0	-1	0	0	1	3	4											
Q Gln	-5	-1	-1	0	0	-1	1	2	2	4										
H His	-3	-1	-1	0	-1	-2	2	1	1	3	6									
R Arg	-4	0	-1	0	-2	-3	0	-1	-1	1	2	6								
K Lys	-5	0	0	-1	-1	-2	1	0	0	1	0	3	5							
M Met	-5	-2	-1	-2	-1	-3	-2	-3	-2	-1	-2	0	0	6						
I Ile	-2	-1	0	-2	-1	-3	-2	-2	-2	-2	-2	-2	-2	2	5					
L Leu	-6	-3	-2	-3	-2	-4	-3	-4	-3	-2	-2	-3	-3	4	2	6				
V Val	-2	-1	0	-1	0	-1	-2	-2	-2	-2	-2	-2	-2	2	4	2	4			
F Phe	-4	-3	-3	-5	-5	-5	-4	-6	-5	-5	-2	-4	-5	0	1	2	-1	9		
Y Tyr	0	-3	-3	-5	-3	-5	-2	-4	-4	-4	0	-4	-4	-2	-1	-1	-2	7	10	
W Trp	-8	-2	-5	-6	-6	-7	-4	-7	-7	-5	-3	2	-3	-4	-5	-2	-6	0	0	17

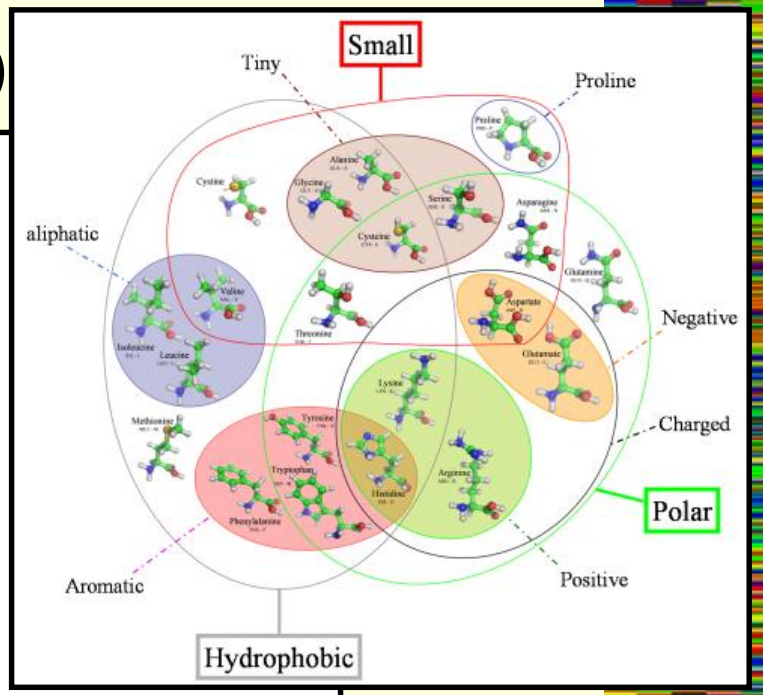
Small

Acid & Amide

Basic

Hydrophobic

Aromatic



Scoring Systems

PAM Matrices

- ***Problems with the PAM approach:***
  - Not all positions are the same, e.g., internal vs external
    - Evolutionary rates vary greatly within a sequence
    - Each position has a unique three dimensional environment
    - Environment changes over evolutionary time as surrounding residues change
  - The most mutable positions were inadvertently selected as the basis for the calculation
    - proteins change more rapidly at the least constrained positions and most slowly at buried “core” positions

Scoring Systems

BLOSUM (BLOcks SUbstitution Matrix)

- ***Based on PROSITE signatures***
  - Signatures are short expressions like C-X-X-C-X-X-X-S-T
- ***Locally align sequences to each signature to get "blocks“***
- ***Blocks are locally conserved regions, i.e., more constrained regions likely to be related to structure/function***
- ***Blocks contain sequences at all different evolutionary distances and may be highly biased (e.g., many identical sequences)***

Scoring Systems

BLOSUM Matrices (Ch 5.1)

- ***Dealing with bias and distance***
  - Cluster all sequences with less than X% identities
    - Clustered sequences count as 1 sequence
    - if X is 100% it simply removes identical sequences
    - if X < 100% it reduces the weight on closely related sequences
  - Calculate substitution frequencies and log-odds matrix
- ***This gives a BLOSUM X table***
  - BLOSUM 62 - sequences greater than 62% identical are clustered
  - BLOSUM 80 - sequences greater than 80% identical are clustered