Bioinformatics
(GLOBEX, Summer 2015)
Pairwise sequence alignment

- Substitution score matrices, PAM, BLOSUM
- Needleman-Wunsch algorithm (Global)
- Smith-Waterman algorithm (Local)
- BLAST (local, heuristic)
- E-value (score significance)

Sequence Alignment

Motivation

- Sequence assembly: reconstructing long DNA sequences from overlapping sequence fragments
- Annotation: assign functions to newly discovered genes
  - Raw genomic (DNA) sequences $\rightarrow$ coding sequences (CDS), candidate for genes $\rightarrow$ protein sequence $\rightarrow$ function
  - Evolution: mutation $\rightarrow$ sequence diversity (versus homology) $\rightarrow$ (new) phenotype ?
  - Basis for annotation: sequence similarity $\rightarrow$ sequence homology $\rightarrow$ same function
    - Caveat: homology can only be inferred, not affirmed, since we can not rewind to see how evolution actually happened.

Ancestral sequence: `ACGTACGT`

After 9540 generations (del: 0.0001, ins: 0.001, trans_mut: 0.00008, transv_mut: 0.00002)

Sequence1: `ACACGGTCCTAATAATGGCC`

Sequence2: `CAGGAAGATCTTAGTTC`

True history:

```
--ACG-T-A---CG-T----
ACACGGTCCTAATAATGGCC

---AC-GTA-C--G-T--
CAG-GAAGATCTTAGTTC
```

Alignment that reflects the true history:

```
Seq1: -AC**AC**-**G**G**T**CCTAAT--AA**T**GGCC
Seq2: CAG-GAA-G-AT--CTTAGTTC--
```

Alignment algorithms

- What is an alignment?

  A one-to-one matching of two sequences so that each character in a pair of sequences is associated with a single character of the other sequence or with a null character (gap). Alignments are often displayed as two rows with an optional third row in between pointing out regions of similarity.
- Example:

```
>gi|7434520|pir||G64632 acetate kinase - Helicobacter pylori (strain 26695)
          Length = 388

 Score = 35.8 bits (81), Expect = 0.10
 Identities = 21/51 (41%), Positives = 29/51 (56%), Gaps = 2/51 (3%)

Query: 1  VLVLNCGSSSLKFAIIDAVNGEEYLSGLAECF--HLPEARIKWKMDGNKQE 49
          +LVLN GSSS+KF + D    +   SGLAE     + + +IK  +  N QE
Sbjct: 3  ILVLNLGSSSIKFKLFDMKENKPLASGLAEKIGEEIGQLKIKSHLHHNDQE 53
```

- Types of alignment:
  - pairwise vs multiple;
  - global vs local
- Algorithms
  - Rigorous
  - heuristic

Substitution Score matrix

- Alignments are used to reveal homologous proteins/genes
- Substitution scores are used to assess how *good* the alignments of a pair of residues are.
- Under the assumption that each mutation (i.e., *del*etion, *ins*ertion, and *sub*stitution) is independent, the total score of an alignment is the sum of scores at each position.
- Substitution score matrix is a 20 x 20 matrix that gives the score for every pair of amino acids.
- The ways to derive a substitution score matrix.
  - *Ad hoc*
  - Physical/chemical properties of amino acids
  - Statistical

PAM matrices (Margaret Dayhoff, 1978)

- point accepted mutation or percent accepted mutation
- unit of measurement of evolutionary divergence between two amino acid sequences
- substitute matrices (scoring matrices)

1 PAM = one accepted point-mutation event per one-hundred amino acids

PAM matrix is a 20 by 20 matrix, and each element p_{ij} represents the expected evolutionary exchange between the two corresponding amino acids for sequences that are a specific number of PAM units diverged. That is,

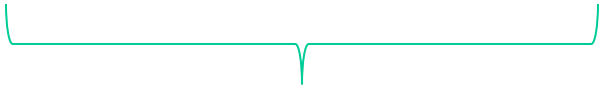
$$p_{ij} = \log[f(i,j)/f(i)f(j)]$$

where f(i) and f(j) are the frequencies that amino acids A_i and A_j appear in the sequences, and f(i,j) the frequency that A_i and A_j are aligned.

PAM1 was manually constructed from sequences that are highly similar (one mutation per 100 amino acids, to be exact) and therefore are easily aligned.

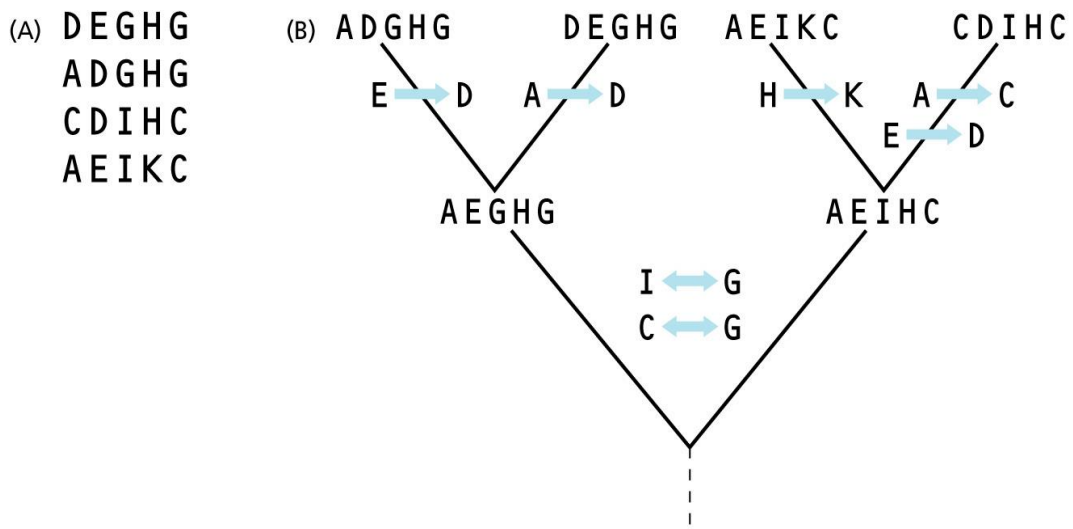
Assuming constant mutation rate, PAMn is constructed by multiplying PAM1 to itself n times. E.g.,

PAM50 = PAM1 x PAM1 x … x PAM1.



50 times

Schematic illustration of constructing substitution score matrix



(C)

	A	C	D	E	G	H	I	K
A		1	1					
C	1				1			
D	1			2				
E			2					
G		1					1	
H								1
I					1			
K						1		

$$p_{ij} = \log[f(i,j)/f(i)f(j)]$$

PAM 250

C	12																		
S	0	2																	
T	-2	1	3																
P	-3	1	0	6															
A	-2	1	1	1	2														
G	-3	1	0	-1	1	5													
N	-4	1	0	-1	0	0	2												
D	-5	0	0	-1	0	1	2	4											
E	-5	0	0	-1	0	0	1	3	4										
Q	-5	-1	-1	0	0	-1	1	2	2	4									
H	-3	-1	-1	0	-1	-2	2	1	1	3	6								
R	-4	0	-1	0	-2	-3	0	-1	-1	1	2	6							
K	-5	0	0	-1	-1	-2	1	0	0	1	0	3	5						
M	-5	-2	-1	-2	-1	-3	-2	-3	-2	-1	-2	0	0	6					
I	-2	-1	0	-2	-1	-3	-2	-2	-2	-2	-2	-2	-2	2	5				
L	-6	-3	-2	-3	-2	-4	-3	-4	-3	-2	-2	-3	-3	4	2	6			
V	-2	-1	0	-1	0	-1	-2	-2	-2	-2	-2	-2	-2	2	4	2	4		
F	-4	-3	-3	-5	-4	-5	-4	-6	-5	-5	-2	-4	-5	0	1	2	-1	9	
Y	0	-3	-3	-5	-3	-5	-2	-4	-4	-4	0	-4	-4	-2	-1	-1	-2	7	10
W	-8	-2	-5	-6	-6	-7	-4	-7	-7	-5	-3	2	-3	-4	-5	-2	-6	0	0 17

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W

BLOSUM matrices [Steven and Jorja Henikoff]

- BLOSUM x matrix is a 20 by 20 matrix. Its elements are defined like those of PAM matrices but the frequencies are collected from sequences in BLOCKs database that are less than x percent identical (generally x is between 50 and 80).

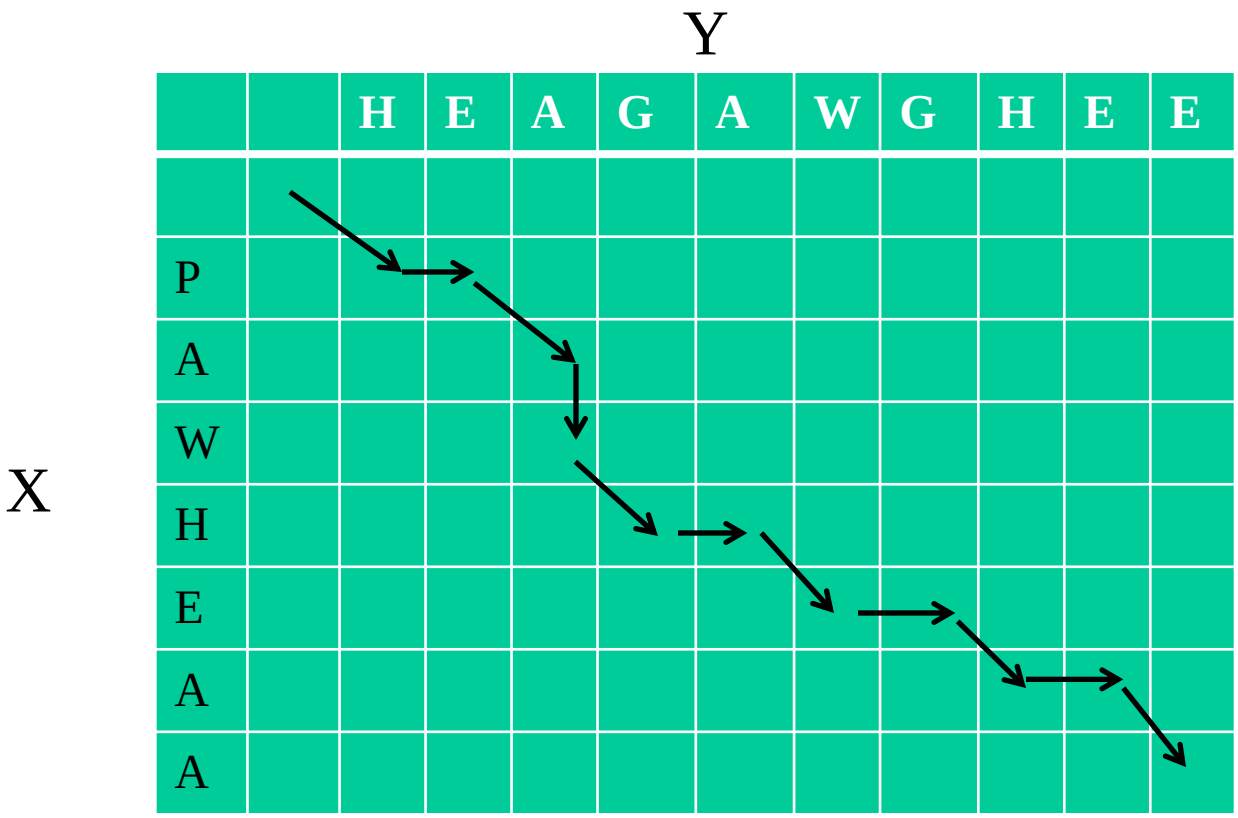
- By their construction, BLOSUM matrices are believed to be more effectively detect distant homology.

- Taking the place of PAM 250, BLOSUM 62 is now the default matrix used in database search.

BLOSUM50

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	5	-2	-1	-2	-1	-1	-1	0	-2	-1	-2	-1	-1	-3	-1	1	0	-3	-2	0
R	-2	7	-1	-2	-4	1	0	-3	0	-4	-3	3	-2	-3	-3	-1	-1	-3	-1	-3
N	-1	-1	7	2	-2	0	0	0	1	-3	-4	0	-2	-4	-2	1	0	-4	-2	-3
D	-2	-2	2	8	-4	0	2	-1	-1	-4	-4	-1	-4	-5	-1	0	-1	-5	-3	-4
C	-1	-4	-2	-4	13	-3	-3	-3	-3	-2	-2	-3	-2	-2	-4	-1	-1	-5	-3	-1
Q	-1	1	0	0	-3	7	2	-2	1	-3	-2	2	0	-4	-1	0	-1	-1	-1	-3
E	-1	0	0	2	-3	2	6	-3	0	-4	-3	1	-2	-3	-1	-1	-1	-3	-2	-3
G	0	-3	0	-1	-3	-2	-3	8	-2	-4	-4	-2	-3	-4	-2	0	-2	-3	-3	-4
H	-2	0	1	-1	-3	1	0	-2	10	-4	-3	0	-1	-1	-2	-1	-2	-3	2	-4
I	-1	-4	-3	-4	-2	-3	-4	-4	-4	5	2	-3	2	0	-3	-3	-1	-3	-1	4
L	-2	-3	-4	-4	-2	-2	-3	-4	-3	2	5	-3	3	1	-4	-3	-1	-2	-1	1
K	-1	3	0	-1	-3	2	1	-2	0	-3	-3	6	-2	-4	-1	0	-1	-3	-2	-3
M	-1	-2	-2	-4	-2	0	-2	-3	-1	2	3	-2	7	0	-3	-2	-1	-1	0	1
F	-3	-3	-4	-5	-2	-4	-3	-4	-1	0	1	-4	0	8	-4	-3	-2	1	4	-1
P	-1	-3	-2	-1	-4	-1	-1	-2	-2	-3	-4	-1	-3	-4	10	-1	-1	-4	-3	-3
S	1	-1	1	0	-1	0	-1	0	-1	-3	-3	0	-2	-3	-1	5	2	-4	-2	-2
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	2	5	-3	-2	0
W	-3	-3	-4	-5	-5	-1	-3	-3	-3	-3	-2	-3	-1	1	-4	-4	-3	15	2	-3
Y	-2	-1	-2	-3	-3	-1	-2	-3	2	-1	-1	-2	0	4	-3	-2	-2	2	8	-1
V	0	-3	-3	-4	-1	-3	-3	-4	-4	4	1	-3	1	-1	-3	-2	0	-3	-1	5

Example: Align HEAGAWGHEE and PAWHEAE.



Any path from upper left corner to lower right corner gives rise to an alignment: diagonal step → align two letters; vertical step → align letter in sequence X to "-"; horizontal step → align letter in sequence Y to "-"

```
HEA-GAWGHEE
P-AWH-E-A-A
```

Example: Align HEAGAWGHEE and PAWHEAE.

```
HEA-GAWGHEE
P-AWH-E-A-A
```

Similarity measured using BLOSUM50 and gap penalty -8:

Score = S(H,P) + S(E,-) + S(A,A) + S(-,W) + S(G,H) + S(A, -) + S(W, E) + S(G,-)
+S(H,A) + S(E,-) + S(E,A)

= -2 -8 +5 -8 -2 -8 -3 -8 -2 -8 -1
= -46

How many possible alignments?

How to find the best alignment?

- brute-force
- Dynamic Programming

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	5	-2	-1	-2	-1	-1	-1	0	-2	-1	-2	-1	-1	-3	-1	1	0	-3	-2	0
R	-2	7	-1	-2	-4	1	0	-3	0	-4	-3	3	-2	-3	-3	-1	-1	-3	-1	-3
N	-1	-1	7	2	-2	0	0	0	1	-3	-4	0	-2	-4	-2	1	0	-4	-2	-3
D	-2	-2	2	8	-4	0	2	-1	-1	-4	-4	-1	-4	-5	-1	0	-1	-5	-3	-4
C	-1	-4	-2	-4	13	-3	-3	-3	-3	-2	-2	-3	-2	-2	-4	-1	-1	-5	-3	-1
Q	-1	1	0	0	-3	7	2	-2	1	-3	-2	2	0	-4	-1	0	-1	-1	-1	-3
E	-1	0	0	2	-3	2	6	-3	0	-4	-3	1	-2	-3	-1	-1	-1	-3	-2	-3
G	0	-3	0	-1	-3	-2	-3	8	-2	-4	-4	-2	-3	-4	-2	0	-2	-3	-3	-4
H	-2	0	1	-1	-3	1	0	-2	10	-4	-3	0	-1	-1	-2	-1	-2	-3	2	-4
I	-1	-4	-3	-4	-2	-3	-4	-4	-4	5	2	-3	2	0	-3	-3	-1	-3	-1	4
L	-2	-3	-4	-4	-2	-2	-3	-4	-3	2	5	-3	3	1	-4	-3	-1	-2	-1	1
K	-1	3	0	-1	-3	2	1	-2	0	-3	-3	6	-2	-4	-1	0	-1	-3	-2	-3
M	-1	-2	-2	-4	-2	0	-2	-3	-1	2	3	-2	7	0	-3	-2	-1	-1	0	1
F	-3	-3	-4	-5	-2	-4	-3	-4	-1	0	1	-4	0	8	-4	-3	-2	1	4	-1
P	-1	-3	-2	-1	-4	-1	-1	-2	-2	-3	-4	-1	-3	-4	10	-1	-1	-4	-3	-3
S	1	-1	1	0	-1	0	-1	0	-1	-3	-3	0	-2	-3	-1	5	2	-4	-2	-2
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	2	5	-3	-2	0
W	-3	-3	-4	-5	-5	-1	-3	-3	-3	-3	-2	-3	-1	1	-4	-4	-3	15	2	-3
Y	-2	-1	-2	-3	-3	-1	-2	-3	2	-1	-1	-2	0	4	-3	-2	-2	2	8	-1
V	0	-3	-3	-4	-1	-3	-3	-4	-4	4	1	-3	1	-1	-3	-2	0	-3	-1	5

Needleman-Wunsch algorithm (Global Pairwise optimal alignment, 1970)

To align two sequences x[1...n] and y[1...m],

i) if x at *i* aligns with y at *j*, a score $s(x_i, y_j)$ is added; if either x_i or y_j is a gap, a score of d is subtracted (penalty).

ii) The ***best*** score up to (i,j) will be

$$F(i,j) = \max \{ F(i-1, j-1) + s(x_i, y_j),$$

$$F(i-1,j) - d, \quad \text{// gap in y}$$

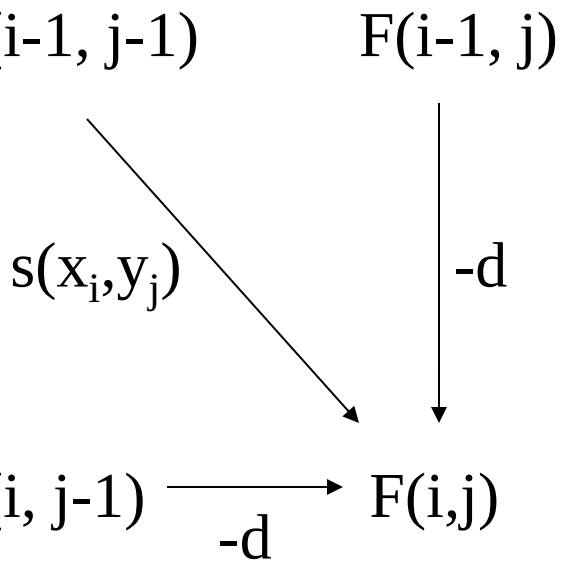
$$F(i, j-1) - d \quad \text{// gap in x}$$

$$\}$$

Needleman-Wunsch (cont'd)

iii) Tabular computing to get F(i,j) for all 1<i<n and i<j<m

Draw a diagram:



By definition, F(n,m) gives the best score for an alignment of x[1...n] and y[1...m].

iv) Trace-back

To find the alignment itself, we must find the path of choices (in applying the formulae of ii) when tabular computing that led to this final value.

> Vertical move is gap in the column sequence.
> Horizontal move is gap in the row sequence.
> Diagonal move is a match.

Example: Align HEAGAWGHEE and PAWHEAE.

Use BLOSUM 50 for substitution matrix and d=-8 for gap penalty.

		H	E	A	G	A	W	G	H	E	E
	0	**-8**	**-16**	-24	-32	-40	-48	-56	-64	-72	-80
P	-8	-2	-9	**-17**	**-25**	-33	-42	-49	-57	-65	-73
A	-16	-10	-3	-4	-12	**-20**	-28	-36	-44	-52	-60
W	-24	-18	-11	-6	-7	-15	**-5**	**-13**	-21	-29	-37
H	-32	-14	-18	-13	-8	-9	-13	-7	**-3**	-11	-19
E	-40	-22	-8	-16	-16	-9	-12	-15	-7	**3**	-5
A	-48	-30	-16	-3	-11	-11	-12	-12	-15	**-5**	2
E	-56	-38	-24	-11	-6	-12	-14	-15	-12	-9	**1**

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	5	-2	-1	-2	-1	-1	-1	0	-2	-1	-2	-1	-1	-3	-1	1	0	-3	-2	0
R	-2	7	-1	-2	-4	1	0	-3	0	-4	-3	3	-2	-3	-3	-1	-1	-3	-1	-3
N	-1	-1	7	2	-2	0	0	0	1	-3	-4	0	-2	-4	-2	1	0	-4	-2	-3
D	-2	-2	2	8	-4	0	2	-1	-1	-4	-4	-1	-4	-5	-1	0	-1	-5	-3	-4
C	-1	-4	-2	-4	13	-3	-3	-3	-3	-2	-2	-3	-2	-2	-4	-1	-1	-5	-3	-1
Q	-1	1	0	0	-3	7	2	-2	1	-3	-2	2	0	-4	-1	0	-1	-1	-1	-3
E	-1	0	0	2	-3	2	6	-3	0	-4	-3	1	-2	-3	-1	-1	-1	-3	-2	-3
G	0	-3	0	-1	-3	-2	-3	8	-2	-4	-4	-2	-3	-4	-2	0	-2	-3	-3	-4
H	-2	0	1	-1	-3	1	0	-2	10	-4	-3	0	-1	-1	-2	-1	-2	-3	2	-4
I	-1	-4	-3	-4	-2	-3	-4	-4	-4	5	2	-3	2	0	-3	-3	-1	-3	-1	4
L	-2	-3	-4	-4	-2	-2	-3	-4	-3	2	5	-3	3	1	-4	-3	-1	-2	-1	1
K	-1	3	0	-1	-3	2	1	-2	0	-3	-3	6	-2	-4	-1	0	-1	-3	-2	-3
M	-1	-2	-2	-4	-2	0	-2	-3	-1	2	3	-2	7	0	-3	-2	-1	-1	0	1
F	-3	-3	-4	-5	-2	-4	-3	-4	-1	0	1	-4	0	8	-4	-3	-2	1	4	-1
P	-1	-3	-2	-1	-4	-1	-1	-2	-2	-3	-4	-1	-3	-4	10	-1	-1	-4	-3	-3
S	1	-1	1	0	-1	0	-1	0	-1	-3	-3	0	-2	-3	-1	5	2	-4	-2	-2
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	2	5	-3	-2	0
W	-3	-3	-4	-5	-5	-1	-3	-3	-3	-3	-2	-3	-1	1	-4	-4	-3	15	2	-3
Y	-2	-1	-2	-3	-3	-1	-2	-3	2	-1	-1	-2	0	4	-3	-2	-2	2	8	-1
V	0	-3	-3	-4	-1	-3	-3	-4	-4	4	1	-3	1	-1	-3	-2	0	-3	-1	5

HEAGAWGHE-E

--P-AW-HEAE

Time complexity: O(nm)

Space complexity: O(nm)

Big-O notation:

f(x) = O(g(x)) => f is upper bound by g

f(x) = Ω(g(x)) => f is lower bound by g

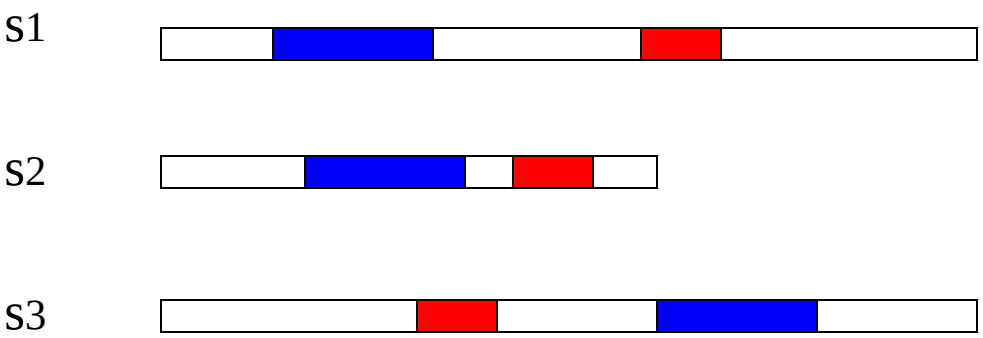
f(x) = Θ(g(x)) => f is bound to g within constant factors

Local pairwise optimal alignment

why need local alignment (vs global)?

- mosaic structure (functioning domains) of proteins, which may be caused by in-frame exchange of whole exons, or alternative splicing)

e.g., are these three sequences similar or not?



Local alignment

- Naive algorithm:
  - there are $\Theta(n^2 m^2)$ pairs of substrings; to align each pair as a global alignment problem will take O(nm); the optimal local alignment will therefore take $O(n^3 m^3)$.

- **Smith-Waterman** algorithm (dynamic programming)
  recurrence relationship

$$F(i,j) = \max \{ 0, F(i-1, j-1) + s(x_i, y_j), F(i-1,j) - d, F(i, j-1) - d \}$$

Notes: 1) For this to work, the random match model must have a negative score. Why?

2) The time complexity of Smith-Waterman is $\Theta(n\ m)$.

Example: Align HEAGAWGHEE and PAWHEAE.

Use BLOSUM 50 for substitution matrix and d=-8 for gap penalty.

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	**0**	0	0	0	0	0	0
A	0	0	0	5	0	**5**	0	0	0	0	0
W	0	0	0	0	2	0	**20**	**12**	0	0	0
H	0	10	2	0	0	0	12	18	**22**	14	6
E	0	2	16	8	0	0	4	10	18	**28**	20
A	0	0	8	21	13	5	0	4	10	20	27
E	0	0	0	13	18	12	4	0	4	16	26

AWGHE

AW-HE

Gap penalties

- Linear

$$\gamma(g) = - g\, d$$

where g is the gap length and d is the penalty for a gap of one base

- Affine

$$\gamma(g) = - d - (g-1)e$$

where d is gap-open penalty and e, typically smaller than d, is gap-extension penalty. Such a distinction is mainly to simulate the observation in alignments: gaps tend to be in a stretch.

Note: gap penalty is a sort of gray area due to less knowledge about gap distribution.

Heuristic alignment algorithms

- motivation: speed

    sequence DB ~ O(100,000,000) basepair

    query sequence 1000 basepair

    O(nm) time complexity => 10^{11} matrix cells in dynamic programming table

    if 10,000,000 cells/second => 10000 seconds ~ 3 hours.

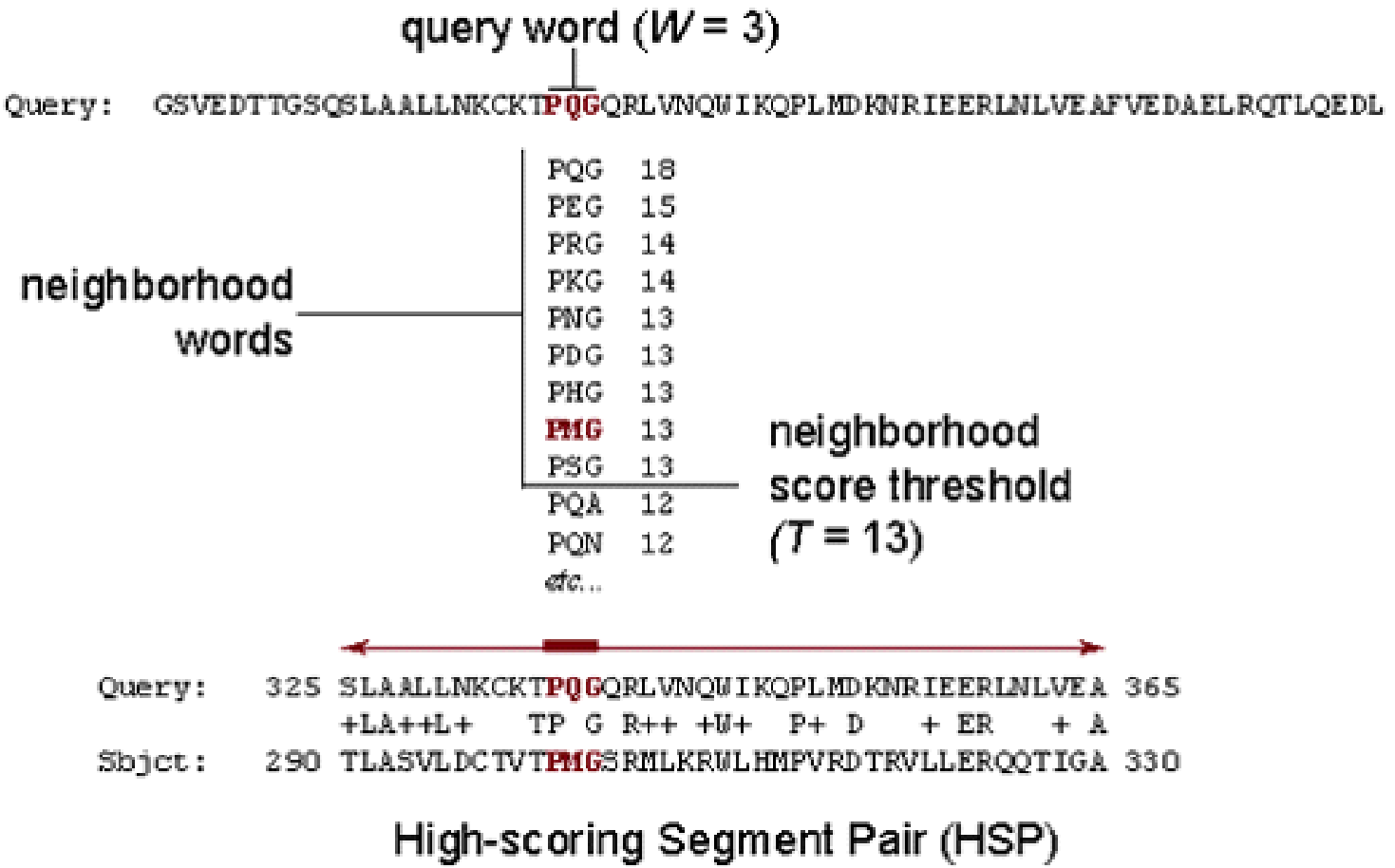
    O(n+m) time => ~ 10 seconds

- heuristic versus rigorous

Basic **L**ocal **A**lignment **S**earch **T**oolkit [Altschul et al, 1990]

1. A list of neighborhood words of fixed length (3 for protein and 11 for DNA) that match the query with score > a threshold.
2. Scan the database sequences and look for words in the list; once find a spot, try a "hit extension" process to extend the possible match as an ungapped alignment in both directions, stopping at the maximum scoring extension.

The BLAST Search Algorithm



Variants of BLAST search

- BLASTP: protein vs. protein
- BLASTN: nucleotide vs. nucleotide
- BLASTX: nucleotide (translated to protein) vs. protein
- TBLASTN: protein vs. nucleotide (translated to protein)
- TBLASTX: nucleotide (translated to protein) vs. nucleotide (translated to protein)

Note: Since proteins are strings of 20 alphabets the odds of having false positive matches is significantly lower than that of DNA sequences, which are strings of 4 alphabets.

Significance of scores

Goals for sequence alignments:

(1) whether and

(2) how two sequences are related.

It is rare that you have just two particular sequences to compare. More often, you have one query sequence and a large database of sequences.

Database searching: find all sequences in the database that are related to the query sequence.

Solution:

(1) For each sequence in the database, use Smith-Waterman/FASTA/BLAST to align with the query sequence and return the score of the optimal alignment.

(2) Rank the sequences by the score.

Q: how good is a score?

Score statistics

Karlin & Altschul 1990

Y.K. Yu & T. Hwa, "Statistical significance of probabilistic sequence alignment and related local hidden Markov models", J. Computational Biology 8(2001)249-282.

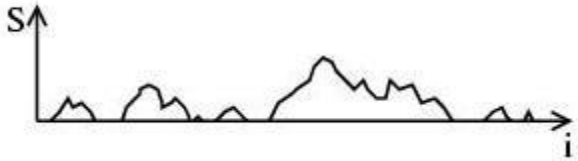
-The score of an ungapped alignment is

$$H_{i,j} = \max\{H_{i-1,j-1} + s(x_i,y_i), 0\}$$

- $\sum_{a,b \in \text{alphabet}} s(a,b)p(a)p(b) < 0 \Rightarrow$ most regions receive zero score.

-The scores of individual sites are independent.

-The landscape of non-zero regions are "islands" in the sea.



-The optimal alignment score S is the global maximum of these island peaks: $S = \max\{\sigma_1, \sigma_2, \sigma_3, \ldots \sigma_\kappa\}$

The probability that the maximum S is smaller than x is

$$P(S < x) = \prod_i [1 - Pr(\sigma_i > x))] \rightarrow \exp[- \kappa e^{-\lambda x}] \text{ when } \kappa \rightarrow \infty.$$

This is a form of **Extreme Value Distribution**.

p-value = probability of at least one sequence scoring with S > x in the given database.

$$P(S > x) = 1- \exp[- \kappa e^{-\lambda x}].$$

E-value = expected number of matches with scores better than S in a database search.

$$E(S) = kmn e^{-\lambda S}.$$

Notes:

- All of the above discussions only applicable to local alignments.
- For gapped local alignments, the same statistics are believed to apply, although not proved.
- The trick is to learn parameters λ and K. These values depend upon the substitution matrix and sequence compositions, and can be estimated from randomly generated data.
- Score statistics for global alignments are not well known.

Q: What is a bit score in the blast search result?

A: The bit score is defined as $S' = (\lambda S - \ln K)/\ln 2$

it is then convenient to calculate the e-value

$$E(S) = mn\, 2^{-S'}$$

```
>qi|7434520|pir||G64632 acetate kinase - Helicobacter pylori (strain 26695)
          Length = 388

 Score = 35.8 bits (81), Expect = 0.10
 Identities = 21/51 (41%), Positives = 29/51 (56%), Gaps = 2/51 (3%)

Query: 1  VLVLNCGSSSLKFAIIDAVNGEEYLSGLAECF--HLPEARIKWKMDGNKQE 49
          +LVLN GSSS+KF + D    +   SGLAE     + + +IK  +  N QE
Sbjct: 3  ILVLNLGSSSIKFKLFDMKENKPLASGLAEKIGEEIGQLKIKSHLHHNDQE 53
```