Sequence Alignments and Similarity Searching

Data Analysis in Genome Biology
GEN242

Thomas Girke

April 12, 2016

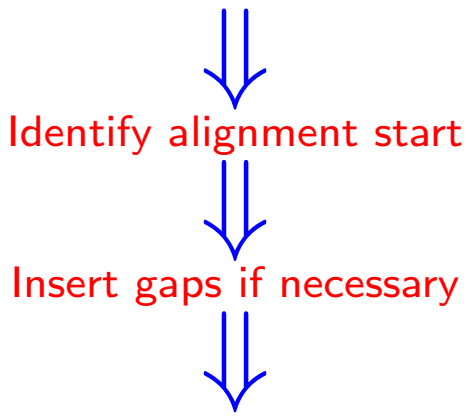
Outline

Illustration of Sequence Alignment Process

Goal: maximize number of identical and similar residues in columns of alignment.

Unaligned Sequences

```
P10632    LKNLNTTAVFMPFSAGKRICAGEGLARMELFGGLFLTTILQNFNLKSVDD
P08686    LAFGCGARVCLGEPLARLELFVVLTRLLQAFTLLPSGD
```



Aligned Sequences

```
P10632      LKNLNTTAVFMPFSAGKRICAGEGLARMELFGGLFLTTILQNFNLKSVDD
P08686      ..........LAFGCGARVCLGEPLARLELF..VVLTRLLQAFTLLPSGD
consensus   ............F..G.R.C.GE.LAR.ELF....LT..LQ.F.L....D
```

logo

Why Comparing Sequences?

Background

- The evolution of biological sequences is mainly driven by gene duplications, point mutations, insertions and deletions.
- Alignment algorithms are **the** central tool to detect these events and to perform sequence similarity analyses in general.

Utilities

- Functional analyses:
  - $\Rightarrow$ Conserved sequence regions are functionally important.
- Evolutionary analyses:
  - $\Rightarrow$ Sequence divergence patters can be used to reconstruct their phylogenetic relationships.
- Mutation and SNP analyses
- Comparative genomics
- Sequence similarity searching is based on alignment methods.
- Many other utilities

Important Terminology

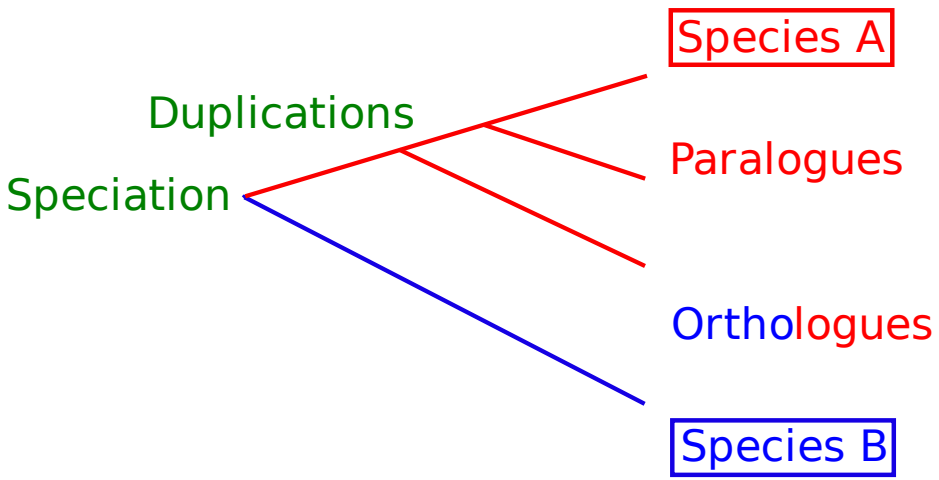
Sequence Identity: often expressed in percent identical residues

Sequence Similarity: often percent of identical and similar residues

Homologous Sequences: evolved from a common ancestor sequence

Orthologous Sequences: homologous sequences from different species

Paralogous Sequences: homologous sequences within one species (gene duplications)



Outline

Sequence Alignments

- Matrix representation of similarities between sequences to identify functional, structural, or evolutionary relationships between them.
- In an alignment the sequences are organized in rows, and their residues with identical or similar properties are arranged in columns.
- Gaps are often introduced to maximize the alignment of similar residues in the same columns.
- High quality alignments arrange the sequences in columns with as many high scoring pairs as possible, while minimizing the cost of unrelated residue pairs and gaps.

Similarity Concepts

- Pairwise Alignments
  - Dot plots
  - Global alignment
  - Local alignment
  - Many additional specialty alignments
- Multiple Alignments

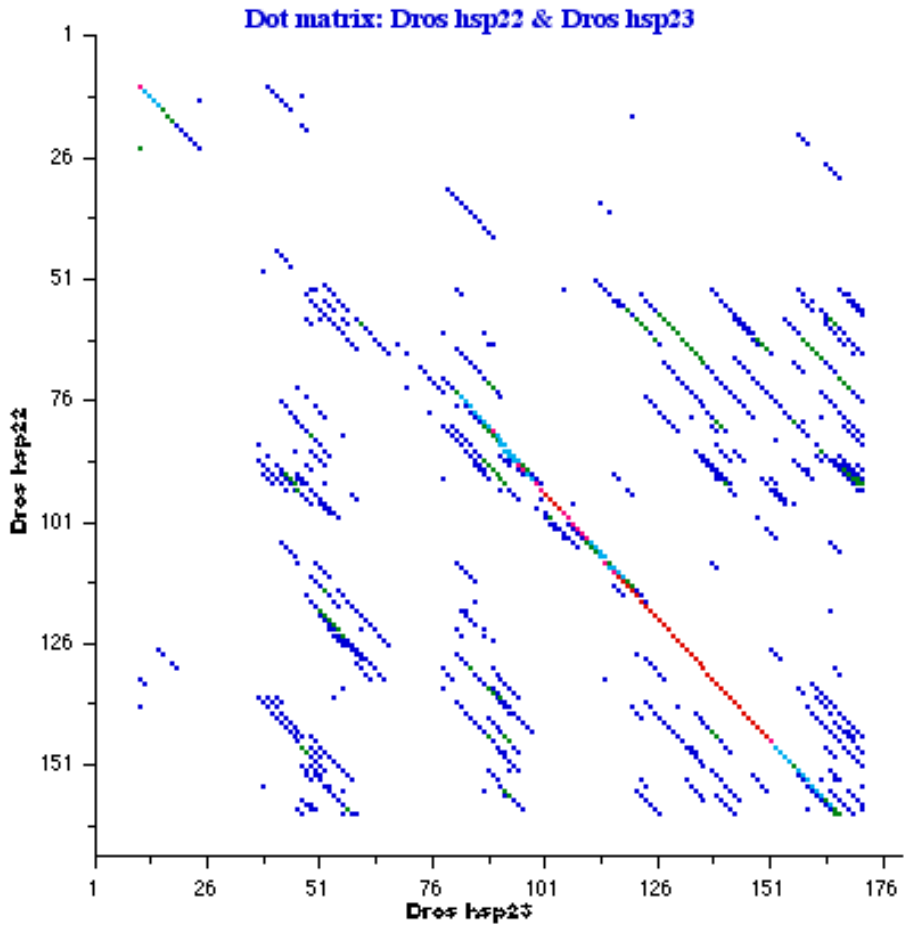
Why Gapped Alignments?

Sequences evolve by complex mutation processes

- Gene duplications*
- Gene deletions*
- Point mutations
  - Substitutions
  - Insertions*
  - Deletions*

$\Rightarrow$ *require gaps

Dot Matrix: Plot of Pairwise Sequence Similarities



Useful to identify: insertions, deletions and repetitive regions.
Commonly used to compare entire chromosomes.

Sequence Alignment Concepts

- String Matching (lacks gaps of alignment approach)

```
H E A G A W G H E E
      A W G H E
```

- Global Pairwise Alignment

```
H E A G A W G H E - E
      |  |   |  |   |
- - P - A W - H E A E
```

- Local Pairwise Alignment

```
  A W G H E
  |  |   |  |
  A W - H E
```

- Multiple Alignment

```
H E A G A W G H E - E
H D A C A W G H E - E
H D A C - W G H E - E
H D - C S T G H E - E
- - P - A W - H E A E
```

Outline

Important Steps in Pairwise Alignment Process

1. Type of alignment
2. Scoring system
3. Algorithm to find best scoring alignment
4. Statistics to evaluate significance of alignment score

Best Sequence Type for Alignments

Divergent Sequences

If available, use protein sequences, because of higher information content, better scoring system, reliability of alignment, functional constraints, etc.

Similar Sequences

Protein or DNA sequence depending on analysis needs.

Outline

Scoring Parameters for Alignments

Substitution matrix

Empirically determined rates at which one residue in a sequence changes to another residue over time.
These substitution rates are typically expressed as:

Log-Odds Scores

$$s_{i,j} = log \frac{p_i * M_{i,j}}{p_i * p_j} = log \frac{observed\ \ frequency}{expected\ \ frequency}$$

$M_{i,j}$ = probability of AA_i transforming into AA_j; p_i = frequency of AA_i.

Gap Opening Penalty

Penalty score for gap insertion. Often severe value to minimize the number of gaps.

Gap Extension Penalty

Penalty score for gap extension. Often severe value to minimize the length of gaps.

Scoring or Substitution Matrices

BLOSUM (BLOcks of Amino Acid SUbstitution Matrix)

Based on functional model for analyzing divergent protein sequences [Henikoff & Henikoff 1992]. The log-odds scores were obtained from the substitution probabilities in conserved and gap-less regions of protein families in the BLOCKS database - one for each of the possible substitutions of the 20 standard amino acids. Matrices with low values (*e.g.* BLOSUM50) are for divergent sequences, and matrices with high values (*e.g.* BLOSUM80) for more related sequences.

PAM (Point Accepted Mutation Matrix)

Based on evolutionary model for analyzing protein sequences [Dayhoff et al 1978]. The mutations are considered throughout the global alignment in conserved and unconserved regions of many well studied protein families. PAM matrices with higher numbers are for studying evolutionary distant sequences, while PAMs with larger numbers are for more related sequences (opposite in BLOSUM matrices).

Matrices for DNA and RNA alignments

Often simple scoring matrices are used where matches have a positive match score, mismatches a negative mismatch score, and gaps a negative gap penalty.

Substitution Matrix

BLOSUM50 (NCBI Matrix Download)

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

B (Asx): aspartic acid, asparagine; Z (Glx): glutamic acid, glutamine; X (Xaa): 'other' amino acid

Gap Penalties

To build high-quality alignments, it is important to control the number and the length of the gaps that are introduced during the alignment building process. This is achieved by selecting gap penalties for the alignment scoring process which are usually negative values.

Constant gap penalty

- Every gap receives the same penalty independent of its size.

Linear gap penalty

- Linear gap penalties have only parameter (d) which is linear to the length of the gap.
- Disadvantage: the overall penalty for one large gap is the same as for many small gaps that add up to the same length.

Affine gap open and extension penalties [most commonly used!]

- Attempts of overcome the problem of the linear gap penalty by using a gap opening penalty (o) and a gap extension penalty (e). Their values are often set so that gap insertions are discouraged and longer gaps are favored over many short gaps.

Outline

Global Alignment: Needleman-Wunsch Algorithm

- Initial algorithm [Needleman and Wunsch 1970]
- Improved version [Gotoh 1982]

Dynamic Programming Alignment Algorithm

Impossible to calculate all possible alignments

The number of possible alignments:

$$\frac{(2n)!}{(n!)^2} \simeq \frac{2^{2n}}{\sqrt{\pi n}}$$

$n =$ length of both sequences

Solution: dynamic programming algorithm

Algorithm for finding an optimal alignment between two sequences with an additive scoring system.

Main Steps in Dynamic Programming Alignment Algorithms

- Recurrence rules: dynamic programming matrix
- Boundary conditions: gaps, termination and extensions
- Traceback step: optimal alignment

Global Sequence Alignment Algorithm

The dynamic programming approach builds an optimal alignment stepwise by adding solutions of optimal sub-alignments. This is achieved with the following steps:

1.1 Construct dynamic programming matrix $F(i,j)$ where the rows i and columns j represent the residues of the two sequences.

2.1 Fill the matrix from the top left to bottom right with the largest score of three possible substitution solutions. First cell is initialized with $F(0,0) = 0$.

$$F(i,j) = max \begin{cases} F(i-1,j-1) + s(x_i, y_i), \\ F(i-1,j) - d, \\ F(i,j-1) - d. \end{cases}$$

$F(i,j)$ = additive substitution score of each sub-solution
d = gap score (gap penalty)

2.2 Boundary rows and columns are filled with: $F(i,0) = -id$ and $F(0,j) = -jd$.

2.3 Apply operation repeatedly to bottom right corner of each square of four cells (see next slide).

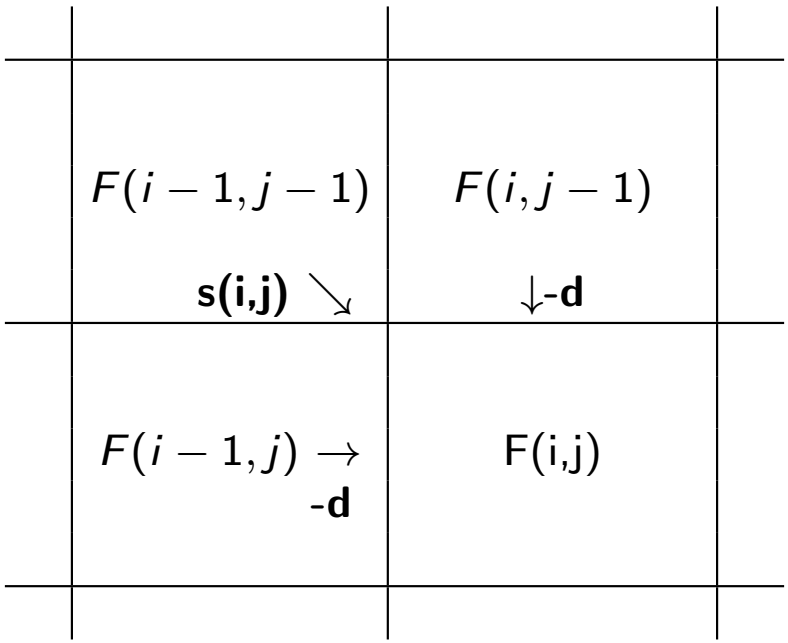
2.4 In each step store a pointer for each cell back to the cell from which $F(i,j)$ was derived.

2.5 Value in final bottom right cell is the final score of the alignment

3.1 Generate alignment by traceback method: starting from final cell move back along the stored pointers and align residues according to movement directions.

- Diagonal movement: align corresponding residues
- Up or left movement: insert gaps accordingly

Three Possibilities : Align, Insert or Delete



$F(i, j)$ = largest of 3 possible solutions

Dynamic Programming Matrix: Global Alignment

Substitution matrix: BLOSUM50
Gap opening and extension penalties: 8

		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 ←	-41 ←	-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**

```
H E A G A W G H E - E
        | |   | |   |
- - P - A W - H E A E
```

Final Score = 1

Complexity of Algorithm

Time and memory cost: $O(nm)$

nm: product of length of two sequences
O: "of order nm"

Outline

Local Alignment: Smith-Waterman Algorithm

- Often more important than global alignment, because related sequences show frequently only local similarities.
- Initial algorithm [Smith and Waterman 1981]
- Improved version [Gotoh 1982]

Local Sequence Alignment Algorithm

Algorithm closely related to global alignment approach, with the following modifications:

1.1 Same as global alignment.

2.1 One more possible solution is added to $F(i,j)$.
If all other solutions are less than zero then $F(i,j)$ will be set to 0:

$$F(i,j) = max \begin{cases} 0, \\ F(i-1,j-1) + s(x_i, y_i), \\ F(i-1,j) - d, \\ F(i,j-1) - d. \end{cases}$$

$F(i,j)$ = additive substitution score of each sub-solution
d = gap score (gap penalty)

2.2 Boundary rows and columns are filled with zeros.

2.3 Taking the value zero corresponds to starting a new alignment.*

2.4 Alignments can start and end anywhere in the matrix.

3.1 Best local alignment: traceback from highest score in matrix until first cell with zero is reached. Value in initial traceback cell is alignment score.

Important requirement: random matches must receive negative values by scoring system, otherwise long unrelated matches would mask significant local matches!

Dynamic Programming Matrix: Local Alignment

Substitution matrix: BLOSUM50
Gap opening and extension penalties: 8

		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**	4	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	6	13	18	12	4	0	4	16	26

```
A W G H E
| |   | |
A W - H E
```

Score of highest ranking alignment = 28

Outline

Additional Algorithms for Pairwise Alignments

Repeat alignment algorithm

- Algorithm for obtaining all non-overlapping local alignments with significant scores.

Maximum overlap match algorithm

- Global alignment algorithm without penalizing overhanging ends like in sequence assembly problem.

Algorithms for allowing long gaps

- Aligning cDNA (no introns) to genomic DNA (introns).

Many more algorithms for specific alignment problems

Repeat Alignment Algorithm

Algorithm for obtaining all non-overlapping local alignments with significant scores.

1.1 Similar to local alignment.

2.1 The matrix is filled starting from $F(i,j) = 0$ following the followings rules:

$$F(i,0) = max \begin{cases} F(i-1,0), \\ F(i-1,j) - T, \end{cases}$$

$$F(i,j) = max \begin{cases} F(i,0), \\ F(i-1,j-1) + s(x_i, y_i), \\ F(i-1,j) - d, \\ F(i,j-1) - d. \end{cases}$$

d = gap score (gap penalty)
T = threshold value*

2.2 $F(i,0)$: is best sum of scores in completed sub-match. It defines ends of matched and unmatched regions.

2.3 $F(i,j)$: defines matches and extensions.

2.4 Total score of all matches is added to matrix at $F(n+1,0)$. It is the final score minus T. If there are no scores greater than T then it will be set to zero.

3.1 Traceback is a global alignment of aligned and unaligned regions (in next slide separated by dots).*

*Note: algorithm obtains all maximal scoring local matches in one pass. Different values of T will provide different results.

Dynamic Programming Matrix: Repeat Alignment

Substitution matrix: BLOSUM50
Gap opening and extension penalties: 8
Threshold T: 20

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	**1**	1	1	1	1	3	**9** ← **9**
P	0	0	0	0	**1**	1	1	1	1	3	9
A	0	0	0	5	0	**6**	1	1	1	3	9
W	**0**	0	0	0	2	0	**21** ←	**13** ←	5	3	9
H	0	**10** ←	2	0	1	1	13	19	**23** ←	15	9
E	0	2	**16** ←	8	1	1	5	11	19	**29**	21
A	0	0	8	**21** ←	13	6	1	5	11	21	28
E	0	0	6	13	18	12 ←	4	1	5	17	27

```
H E A G A W G H E E (continuous sequence)
| | |   | |   | |
H E A . A W - H E . (repeat sequence)
```

Total score: highest score minus T: 29-20 = 9.
Two sub-alignments with scores 1 and 8.

Maximum Overlap Match Algorithm

Global alignment algorithm without penalizing overhanging ends.

1.1 Same as global alignment.

2.1 Recurrence condition is the same as for global alignment:

$$F(i,j) = max \begin{cases} F(i-1,j-1) + s(x_i, y_i), \\ F(i-1,j) - d, \\ F(i,j-1) - d. \end{cases}$$

$F(i,j)$ = additive substitution score of each sub-solution
d = gap score (gap penalty)

2.2 Boundary rows and columns are filled with zeros.

2.3 Final score is maximum value on the bottom of right border.

3.1 Traceback starts from cell with maximum score on right border and continues until top left border is reached.

Dynamic Programming Matrix: Overlap Match

Substitution matrix: BLOSUM50
Gap opening and extension penalties: 8

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	-2	-1	-1	**-2**	-1	-4	-2	-2	-1	-1
A	0	-2	-2	4	-1	**3**	-4	-4	-4	-3	-2
W	0	-3	-5	-4	1	-4	**18** ←	**10** ←	2	6	-6
H	0	10 ←	2 ←	6	-6	-1	10	16	**20** ←	12 ←	4
E	0	2	16 ←	8 ←	0	7	2	8	16	**26**	18
A	0	-2	8	21 ←	13	5 ←	3 ←	2	8	18	**25**
E	0	0	4	13	18	12 ←	4 ←	4	2	14	24

```
G A W G H E E
  | |   | |
P A W - H E A
```

Score of optimal overlap = 25

Summary

- Many variants of dynamic programming algorithms exist. They mainly differ in their:
  - Boundary conditions
  - Recurrence rules
- They are all necessary to solve the wide variety of alignment problems.
- An all-purpose alignment algorithm does not exist.

Outline

Outline

Why Sequence Similarity Searching?

Essential tool for retrieving related sequences from databases by providing protein or DNA sequences as queries.

Applications

- Similarity-function principle to predict gene functions
  - $\Rightarrow$ If the function of a query sequence is unknown, and a sequence similarity search retrieves highly similar sequences of known function, then it is likely that the query sequence has a similar function.
- Principle of relatedness for evolutionary analyses
  - $\Rightarrow$ Sequence similarities can be used to reconstruct their phylogenetic relationships.
  - $\Rightarrow$ For example: identification of sequences with a common ancestors, such as orthologs and paralogs
- Discovery of new genes or proteins.
- Exploring gene and protein structures.
- ...

Outline

SSearch for Searching Sequence Databases

http://fasta.bioch.virginia.edu/fasta/fasta_list.html

- SSearch performs a rigorous Smith-Waterman sequence similarity search of a query sequence against a sequence database.
- It is the iterative version of Smith-Waterman algorithm for pairwise alignments by performing the following computations:
  A. Align and score a query sequence against all members in database using Smith-Waterman algorithm.
  B. Rank search results by score.
- It is one of the most sensitive methods available for sequence similarity searching.
- It is much slower than the BLAST and FASTA search methods.
- Hardware solution: Smith-Waterman searches on FPGAs (field-programmable gate arrays), 1000-2000 acceleration compared to CPUs.

Speed Acceleration by Heuristic Approaches

Rigorous Approaches

- Can guarantee optimum solution to a problem.
- Often impossible to compute because of extreme computation time. For example, impossibility of computing all possible pairwise alignments between two sequences, due to following relationship:

$$\text{Number of possible alignments} \simeq \frac{2^{2n}}{\sqrt{\pi n}}$$

$$n = \text{length of both sequences}$$

Heuristic Approaches

- Overcome computation time limitations by providing approximate rather than complete solutions.
- Optimum solution is not guaranteed.
- Approximate solutions are often of acceptable accuracy.

Outline

BLAST: Basic Local Alignment Search Tool

- Developed by [Altschul et al 1990]
- Most widely used similarity search tool
- Heuristic approach based on Smith-Waterman algorithm
- Finds best local alignments
- Provides statistical significance
- Online, command-line, and network clients

How BLAST Works?

Step 1: Generate lookup hash table of query words

Step 2: Scan database for hits

Step 3: Ungapped extensions of hits

Step 4: Gapped extensions of hits with traceback

Step 5: Rank hits by scoring system

Step 1: Generate Lookup Table for Query

- Create lookup word table of query sequence by a moving window of size w.
- Example of lookup word table of query sequence with $w = 3$:

```
Query:  GTQITVEDLFY
        GTQ
         TQI
          QIT
           ITV
            TVE
             VED
              EDL
               DLF
                LFY
```

- Word size w for proteins: 2 or 3 (3 is default).
- Word size w for BLASTN: min 7 (11 is default).

Step 2: Scan Database for Hits

- Match query lookup table against similar lookup table in database:

```
Query:              ITV                 MSV
Database:  LTV, ITV, MTV, LSV, MSV, IAV, ...
```

Step 3: Ungapped Extensions of Hits

- Once a hit of a query word is found in the database, extend the hit in the sequences in either direction:

Step 4: Gapped Extensions of Hits

- Further extend alignment by gapped alignment with traceback method (dynamic programming):

```
Query:              GTQITVEDL--FY
                    |||||||||  ||
Database:  WHKLCGTQITVEDLAQFY
```

- Keep track of the score by using substitution matrix (*e.g.* BLOSUM50).
- Stop when the score drops below some cutoff.

Step 5: Rank Hits by Scoring System

- BLAST search results in local alignments which are called HSPs (High Similarity Pairs).
- Their scores follow an extreme value distribution (EVD).
- E values are the most relevant scores of a BLAST search result. They are derived from the analysis of the distribution of alignment scores.
- Equation for calculating E values for an HSP:

$$E = Kmne^{-\lambda S}$$

E = number of hits expected from search with scores greater than S
K = scale for search space (constant)
m = size of query sequence
n = size of database
S = score
λ = scale for the specific scoring matrix

- Searches against larger databases give less significant E values than against smaller databases because of n dependency in above formula!

Meaning of E Value in BLAST Searches

The E value (expectation value) expresses the number of different alignments with scores equivalent to or better than S that are expected to occur in a database search by chance.

- The lower the E value, the more significant the score.
- Low E values are almost identical with P values:

E value	P value $(1 - e^{-Evalue})$
10	0.99995460
5	0.99326205
1	0.63212056
0.1	0.09516258
0.001	0.00099950
0.0001	0.0001000

Some More Details

- Nucleotide BLAST looks for exact matches

```
CGTAGCTACGTAGCTACTACTACGTAC
      TACGTAGCTAC
```

- Protein BLAST requires two neighborhood matches

```
GTQITVEDLFYNI
  QIT    FYN
```

- SEQ and DUST programs are used to mask low complexity regions in query sequences

One Important Limitation of BLAST

Alignments that BLAST can't find because of word size limitation of 7 nucleotides for DNA sequences:

```
Query:   ATCTACTACTACTTAGATCGAGCGTACGTGTTGACACACTATCTAC
         |||||| |||||| |||||| |||||| |||||| |||||| ||||
Subject: ATCTACCACTACTGAGATCGTGCGTACATGTTGAAACACTAGCTAC
```

Untranslated and Translated BLAST Tools

Untranslated BLAST Tools

- BLASTN: query DNA vs DNA database
- BLASTP: query protein vs protein database

Translated BLAST Tools

- BLASTX: translated query DNA vs protein database
- TBLASTN: query protein vs translated DNA database
- TBLASTX: translated query DNA vs translated DNA database

BLAST Flavors

Different BLAST Programs: http://www.ncbi.nlm.nih.gov/BLAST

- blastall: command-line collection of BLAST tools
- BLAST: BLASTN, BLASTP, BLASTX, TBLASTN, TBLASTX
- Psi-BLAST: Position-Specific Iterated BLAST
- RPS-BLAST: Reverse Position-Specific BLAST
- Phi-BLAST: Pattern Hit Initiated BLAST
- Mega-BLAST: 10 faster than BLASTN
- BLAST2: pairwise comparisons
- WU-BLAST: Washington University BLAST

Outline

FASTA Database Search Algorithm

http://fasta.bioch.virginia.edu/fasta/fasta_list.html

- Heuristic sequence similarity search approach developed by Pearson and Lipman 1988.
- Identifies in first step all identical words of given length between query and database sequences.
- Identifies diagonals for best word matches.
- Extends word matches to identify best scoring ungapped matches.
- Tries to join ungapped matches by insertion of gaps.
- Highest scoring candidates are aligned with full dynamic programming algorithm

Outline

Selected Software

- Pairwise Alignment Programs
  - Smith-Waterman local alignment: WATER (EMBOSS)
  - BLAST-like local alignments: BLAST2
  - Global alignments: NEEDLE (EMBOSS)
  - ...
- Sequence Similarity Search Programs
  - BLAST and its variants: online and download
  - SSearch
  - FASTA
  - Short read alignment tools
  - ...

Outline

Homework

1. Choice of Sequence Type
   - Which sequence type - amino acid or nucleotide - is more appropriate to search databases for remotely related sequences?
   a. Provide at least three reasons for your decision.
2. Dynamic Programming for Pairwise Alignments
   - Create manually one global and one local alignment for the following two protein sequences using the Needleman-Wusch and Smith-Waterman algorithms:

     ```
     O15528:  PFGFGKRSCMGRRLA
     P98187:  FIPFSAGPRNCIGQK
     ```

   - Use in each case BLOSUM50 as substitution matrix and 8 as gap opening and extension penalties.

   a.-c. Provide in your answer the manually populated dynamic programming matrices, the optimum pairwise alignments created by traceback and their final scores.
3. Alignments with Different Substitution Matrices
   - Load the `Biostrings` package in R, import the two cytochrome P450 sequences O15528 and P98187 from NCBI and create a global alignment with the `pairwiseAlignment` function:

     ```
     > myseq <- readAAStringSet("myseq.fasta", "fasta")
     > (p <- pairwiseAlignment(myseq[[1]], myseq[[2]], type="global",
     substitutionMatrix="BLOSUM50"))

     > writePairwiseAlignments(p)
     ```

   a. Record the scores for the scoring matrices BLOSUM50, BLOSUM62 and BLOSUM80
   b. How and why do the scores differ for the three scoring matrices?

Outline

References and Books

Altschul, S F, Gish, W, Miller, W, Myers, E W, Lipman, D J (1990) Basic local alignment search tool. J Mol Biol, 215: 403-410.
URL http://www.hubmed.org/display.cgi?uids=2231712

Dayhoff, MO, Schwartz, RM, Orcutt, BC (1978) A model of evolutionary change in proteins. Atlas of Protein Sequence and Structure: Vol 5, 345-352.

Gotoh O (1982) An improved algorithm for matching biological sequences. J Mol Biol 162, 705-708.
URL http://www.hubmed.org/display.cgi?uids=7166760

Henikoff, S, Henikoff, JG (1992) Amino Acid Substitution Matrices from Protein Blocks. PNAS 89: 10915-10919.
URL http://www.hubmed.org/display.cgi?uids=1438297

Needleman SB, Wunsch CD (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. J Mol Biol 48, 443-453.
URL http://www.hubmed.org/display.cgi?uids=5420325

Smith TF, Waterman MS (1981) Identification of common molecular subsequences. J Mol Biol 147, 195-197.
URL http://www.hubmed.org/display.cgi?uids=7265238