03 Sequence alignment

03.04 Special-purpose alignment algorithms

- Overlap matches
- Local alignment
- Repeated matches

Overlap matches

(allow ends to slide with no penalty)

- Align strings that either contain each other or overlap
- Similar to global alignment, but gaps at the two ends of each string are not penalized
  - Alignment can start from any cell in the top or left border of the matrix $F(0,j)=0$ $F(j,0)=0$
  - Alignment can end in any cell in the right or bottom border

$$S(X,Y)=\max\begin{cases}\max_i F(i,N)\\ \max_j F(M,j)\end{cases}$$

  - Recursive equation is unchanged

Overlap matches

(allow ends to slide with no penalty)

Example: BLOSUM50, linear gap penalty d=8

GAWGHEE
PAW-HEA

		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

Local alignment

[Smith-Waterman, 1981]

- Looks for best alignment between subsequences of *X* and *Y*
- There are two main differences with respect to Needleman-Wunsch:

1. Negative scores are not allowed

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

1. The score of the best local alignment is the highest score in the matrix (that can be enywhere)

Local alignment
[Smith-Waterman, 1981]

Example: BLOSUM50, linear gap penalty d=8

AWGHE
AW-HE

		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

Repeated matches (asymmetric)

- Sequence *X* contains a *domain*, or *motif*
- We look for multiple local matches of *X* in sequence *Y*, with score higher than a positive threshold *T*

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

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

Repeated matches

HEAGAWGHEE

HEA.AW-HE.

Example: BLOSUM50, linear gap penalty d=8, 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	1	**6**	1	1	1	3	9	
W	**0**	0	0	0	2	1	**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	0	13	18	12	4	1	5	17	27	

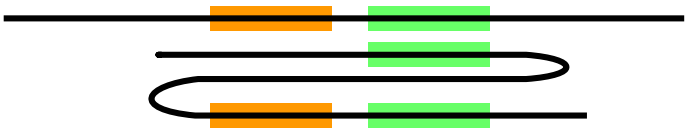
Comparing alignments

Global alignment

Overlap

Local alignment

Repeated matches



03 Sequence alignment

03.05 Heuristic algorithms

- Problem statement
- BLAST
- FASTA

Heuristics

- The issue of performance
  - Dynamic programming requires O(*MN*) steps
  - Data base size (e.g. TrMBL)
    - 314,641,655 amino acids in november 2003
    - Grows very fast
  - Query length: 300 a.a.
  - Complexity: 10^{11} cells
  - Computation time (at 10^7 cells per second): 3 hours per query
- Solution
  - Heuristic techniques trading off accuracy for performance
  - Return not-only the best match to compensate for inaccuracy

BLAST

[Altschul *et al.,* 1990]

Use short high-score matching words as seeds from which to extend the alignment

- Create a table of short words (i.e., 3r, 11bp) that match the query with high score
- Scan the database searching for words in the table
- Extend the matching region (without gaps) stopping at the maximum scoring extension

FASTA

[Pearson and Lipman, 1988]

Search for short high-score matching words arranged on the same diagonal

1. Use a lookup table to locate short matching words (i.e., 2r, 6bp)
2. Search for diagonals with many matching words (this can be done by sorting matches on the difference of their indices *i-j*)
3. Extend the matching seeds in the best diagonals (without gaps)
4. Use gaps to join ungapped regions

FASTA

[Pearson and Lipman, 1988]

Example: BLOSUM50, linear gap penalty *d*=8, *seed_size*=1, *min_score*=6

1. Locate seeds with over threshold score

		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	-1	**5**	0	**5**	-3	0	-2	-1	-1
W	0	-3	-3	-3	-3	-3	**15**	-3	-3	-3	-3
H	0	**10**	0	-2	-2	-2	-3	-2	**10**	0	0
E	0	0	**6**	-1	-3	-1	-3	-3	0	**6**	**6**
A	0	-2	-1	**5**	0	**5**	-3	0	-2	-1	-1
E	0	0	**6**	-1	-3	-1	-3	-3	0	**6**	**6**

FASTA

[Pearson and Lipman, 1988]

Example: BLOSUM50, linear gap penalty *d*=8, *seed_size*=1, *min_score*=6

2. Find best diagonal with multiple supporting seeds

		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	-1	**5**	0	**5**	-3	0	-2	-1	-1	6
W	0	-3	-3	-3	-3	-3	**15**	-3	-3	-3	-3	
H	0	**10**	0	-2	-2	-2	-3	-2	**10**	0	0	
E	0	0	**6**	-1	-3	-1	-3	-3	0	**6**	**6**	16
A	0	-2	-1	**5**	0	**5**	-3	0	-2	-1	-1	
E	0	0	**6**	-1	-3	-1	-3	-3	0	**6**	**6**	21
			6		16				6			

FASTA

[Pearson and Lipman, 1988]

Example: BLOSUM50, linear gap penalty *d*=8, *seed_size*=1, *min_score*=6

3. Maximum ungapped extension

		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	-1	**5**	0	**5**	-3	0	-2	-1	-1
W	0	-3	-3	-3	-3	-3	**15**	-3	-3	-3	-3
H	0	**10**	0	-2	-2	-2	-3	-2	**10**	0	0
E	0	0	**6**	-1	-3	-1	-3	-3	0	**6**	**6**
A	0	-2	-1	**5**	0	**5**	-3	0	-2	-1	-1
E	0	0	**6**	-1	-3	-1	-3	-3	0	**6**	**6**

23

FASTA

[Pearson and Lipman, 1988]

Example: BLOSUM50, linear gap penalty *d*=8, *seed_size*=1, *min_score*=6

4. Possibly add gaps

		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	-1	**5**	0	**5**	-3	0	-2	-1	-1
W	0	-3	-3	-3	-3	-3	**15**	-8	-3	-3	-3
H	0	**10**	0	-2	-2	-2	-3	-2	**10**	0	0
E	0	0	**6**	-1	-3	-1	-3	-3	0	**6**	**6**
A	0	-2	-1	**5**	0	**5**	-3	0	-2	-1	-1
E	0	0	**6**	-1	-3	-1	-3	-3	0	**6**	**6**

28