Smith-Waterman Algorithm

AMPP 0708-Q1
Eduard Ayguade
Juan J. Navarro
Dani Jimenez-Gonzalez

October 4, 2007

1. Introduction
   - Why compare sequences of aminoacids?
   - How to compare sequences? Alignment
   - Scoring the relationships
   - How to find the best alignment?

2. Smith-Waterman Algorithm

Why compare sequences of aminoacids?

- Proteins are made by aminoacid sequences

  t:c g g g t a t c c a a

- Similar sequences of aminoacids → similar protein structures

  t:c g g g t a t c c a a
  s:c c c t a g g t c c c a

- Evolutionary perspective: Mutations?, insertions?, etc.
  - $t_1 = g$ mutated to $s_1 = c$?
  - $s_1 = c$ has been an insertion?
- Some evolution are more important/likely than others

How to compare sequences? Alignment

An alignment of two sequences *t* and *s* must satisfy:

- All symbols (residues) in the two sequences have to be in the alignment, and in the same order they appear in the sequences
- We can align one symbol from one sequence with one from the another
- A symbol can be aligned with a blank ('-')
- Two blanks cannot be aligned

```
t: c g g g t a t c c a a
s: c c c t a g g t c c c a

t: c g g g t a - - t - c c a a
s: c c - t a g g t c c c - a
```

What is the BEST alignment?

Example

```
t: c g g g t a t c c a a
s: c c c t a g g t c c c a
```

What is the BEST alignment?

Example

```
t: c g g g t a t c c a a
s: c c c t a g g t c c c a

t: c g g g t a - - t - c c a a
s: c c c - t a g g t c c c - a
```

What is the BEST alignment?

Example

```
t: c g g g t a t c c a a
s: c c c t a g g t c c c a

t: c g g g t a - - t - c c a a
s: c c c - t a g g t c c c - a

t: c g g g t a - - - t c c a a
s: c c - - c t a g g t c c c a
```

What is the BEST alignment?

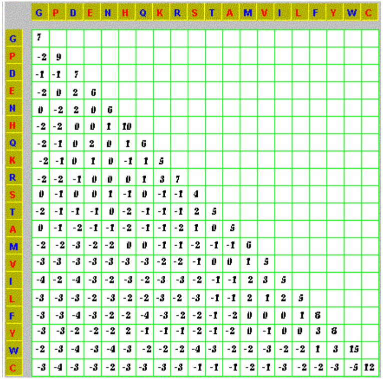
Example

```
t: c g g g t a t c c a a
s: c c c t a g g t c c c a

t: c g g g t a - - t - c c a a
s: c c c - t a g g t c c c - a

t: c g g g t a - - - t c c a a
s: c c - - c t a g g t c c c a

t: c - g g g t a - - t c c a a
s: c c - - c t a g g t c c c a
```

Which is the best?

Scoring the relationships

- Needed a scoring matrix
- We will be able to find a optimal solution for the scoring matrix at hand



	G	P	D	E	N	H	Q	K	R	S	T	A	M	V	I	L	F	Y	W	C
G	7																			
P	-2	9																		
D	-1	-1	7																	
E	-2	0	2	6																
N	0	-2	2	0	6															
H	-2	-2	0	0	1	10														
Q	-2	-1	0	2	0	1	6													
K	-2	-1	0	1	0	-1	1	5												
R	-2	-2	-1	0	0	0	1	3	7											
S	0	-1	0	0	1	-1	0	-1	-1	4										
T	-2	-1	-1	-1	0	-2	-1	-1	-1	2	5									
A	0	-1	-2	-1	-1	-2	-1	-1	-2	1	0	5								
M	-2	-2	-3	-2	-2	0	0	-1	-1	-2	-1	-1	6							
V	-3	-3	-3	-3	-3	-3	-3	-2	-2	-1	0	0	1	5						
I	-4	-2	-4	-3	-2	-3	-2	-3	-3	-2	-1	-1	2	3	5					
L	-3	-3	-3	-2	-3	-2	-2	-3	-2	-3	-1	-1	2	1	2	5				
F	-3	-3	-4	-3	-2	-2	-4	-3	-2	-2	-1	-2	0	0	0	1	8			
Y	-3	-3	-2	-2	-2	2	-1	-1	-1	-2	-1	-2	0	-1	0	0	3	8		
W	-2	-3	-4	-3	-4	-3	-2	-2	-2	-4	-3	-2	-2	-3	-2	-2	1	3	15	
C	-3	-4	-3	-3	-2	-3	-3	-3	-3	-1	-1	-1	-2	-1	-3	-2	-2	-3	-5	12

Figure: BLOSUM scoring matrix, S.

What is the BEST alignment (for that Score Matrix)?

Example

```
t: c g g g t a t c c a a
s: c c c t a g g t c c c a
```

t :	*c*	*g*	*g*	*g*	*t*	*a*	—	—	*t*	—	*c*	*c*	*a*	*a*	
	+12	−3	−3	−1	+5	+5	−1	−1	+5	−1	+12	+12	−1	+5	45
s :	*c*	*c*	*c*	—	*t*	*a*	*g*	*g*	*t*	*c*	*c*	*c*	—	*a*	

t :	*c*	*g*	*g*	*g*	*t*	*a*	—	—	—	*t*	*c*	*c*	*a*	*a*	
	+12	−3	−1	−1	−1	+0	−1	−1	−1	+5	+12	+12	−1	+5	36
s :	*c*	*c*	—	—	*c*	*t*	*a*	*g*	*g*	*t*	*c*	*c*	*c*	*a*	

t :	*c*	—	*g*	*g*	*g*	*t*	*a*	—	—	*t*	*c*	*c*	*a*	*a*	
	+12	−1	−1	−1	−3	+5	+5	−1	−1	+5	+12	+12	−1	+5	47
s :	*c*	*c*	—	—	*c*	*t*	*a*	*g*	*g*	*t*	*c*	*c*	*c*	*a*	

How to find the best alignment?

- Homology search methods begin with DP algorithms
  - Needleman-Wusch: global search
  - Smith-Waterman (SW): local search
- Faster but less sensitive for larger datasets
  - FASTA
  - BLAST
- Optimal spaced seeds of pattern-writer increase
  - Speed and sensitivity
  - Similar to SW
  - Examples: Pattern Hunter and BLAT
    - SW sensitivity
    - BLAST speed

Smith-Waterman Algorithm

- $N \ x \ N$ integer matrix
- N is sequence length (both s and t)
- Compute $M[i][j]$ based on Score Matrix and optimum score compute so far (DP)

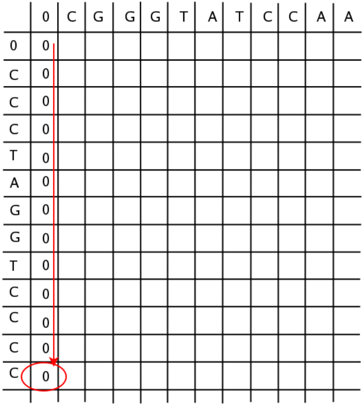
Figure: Computation Matrix alginment, M

	0	C	G	G	G	T	A	T	C	C	A	A
0												
C												
C												
C												
T												
A												
G												
G												
T												
C												
C												
C												
C												

Smith-Waterman Algorithm: Understanding Matrix

Alignment

$$\begin{array}{lcccccccc} t: & - & - & - & - & - & - & - & - \\ s: & c & c & c & t & a & g & g & t \end{array}$$



	0	C	G	G	G	T	A	T	C	C	A	A
0	0											
C	0											
C	0											
C	0											
T	0											
A	0											
G	0											
G	0											
T	0											
C	0											
C	0											
C	0											
C	0											

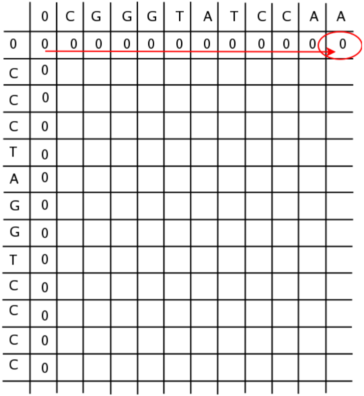
Figure: Aligning *s* to gaps

Smith-Waterman Algorithm: Understanding Matrix

Alignment

$t: \quad c \quad g \quad g \quad g \quad t \quad a \quad t \quad \ldots$

$s: \quad - \quad - \quad - \quad - \quad - \quad - \quad - \quad \ldots$



	0	C	G	G	G	T	A	T	C	C	A	A
0	0	0	0	0	0	0	0	0	0	0	0	0
C	0											
C	0											
C	0											
T	0											
A	0											
G	0											
G	0											
T	0											
C	0											
C	0											
C	0											
C	0											

Figure: Aligning t to gaps

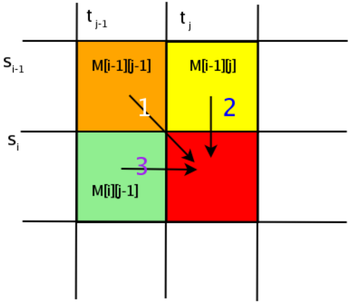
Smith-Waterman Algorithm: How to compute cell score?

How to find $M[i][j]$?
Three ways to finish the alignment of $s_{0..i}$ and $t_{0..j}$

1. Score $\begin{array}{c} s_i \\ t_j \end{array}$

2. Gap in t $\begin{array}{cc} & s_i \\ t_j & - \end{array}$

3. Gap in s $\begin{array}{cc} s_i & - \\ & t_j \end{array}$



Smith-Waterman Algorithm: How to compute cell score?

How to find $M[i][j]$?
Three ways to finish the alignment of $s_{0..i}$ and $t_{0..j}$

1. $M[i-1][j-1] + S[s_i][t_j]$

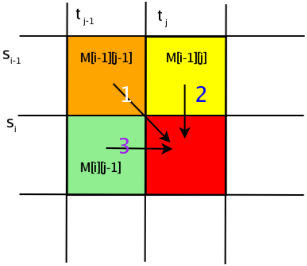
$$\begin{array}{cc} & s_i \\ & t_j \end{array}$$

2. $M[i-1][j] - g$

$$\begin{array}{cc} & s_i \\ t_j & - \end{array}$$

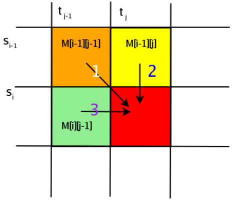
3. $M[i][j-1] - g$

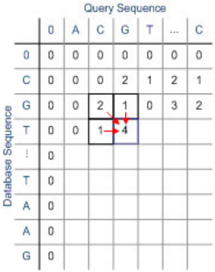
$$\begin{array}{cc} s_i & - \\ & t_j \end{array}$$



Smith-Waterman Algorithm: Scoring Process

Element Computation $M[i][j]$:





Query Sequence
Database Sequence

	0	A	C	G	T	...	C
0	0	0	0	0	0	0	0
C	0	0	0	2	1	2	1
G	0	0	2	1	0	3	2
T	0	0	1	4			
⋮	0						
T	0						
A	0						
A	0						
G	0						

$$M[i][0] = 0 \quad M[0][j] = 0$$

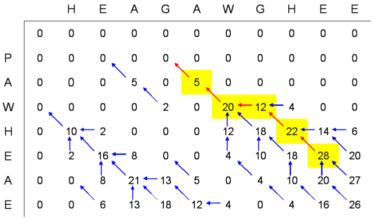
$$M[i][j] = \max \begin{cases} 0 & \\ M[i-1][j-1] + S[s_i][t_j] & \text{if } s_i \ t_j \\ M[i-1][j] - d & \text{if } s_i \text{ -} \\ M[i][j-1] - d & \text{if - } t_j \end{cases}$$

Smith-Waterman Algorithm: Backtracking Process

If we want to find **BEST** local alignment...

- Find $Score_{opt}$ and then traceback

$$Score_{opt} = \max_{i,j=1}^{N} M[i][j]$$



		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

x = AWGHE
y = AW−HE