CS123A Needleman-Wunsch Global Alignment Example

This document contains a detailed step-by-step example of how the initial gap and gap extension version of the Needleman-Wunsch (NW) global alignment algorithm works

1. Needleman-Wunsch Global Alignment Example:

To explain how initial gap and consecutive gap insertion penalties work within the context of computing alignment scores, we begin by defining an equation for the gap penalty, that is

$$g(n_{gap}) = -I - n_{gap}E \quad \text{(Eq-1)}$$

Where n_{gap} is the length of the gap where $n_{gap} \geq 1$, and I is the penalty for inserting the first/initial gap, and E is a positive number that reflects the penalty for inserting one or more gaps after the nucleotide/residue should be higher for some nucleotides/residues than others.

The equation that we will use to calculate the NW scores for each cell in a scoring matrix that will take into account initial gap and consecutive gap insertion penalties is

NOTE #1

Just for your information, there are yet additional variations of Eq-1. For example, there is a version of $g(n_{gap})$ that is defined as

$$g(n_{gap}) = -I - (n_{gap} - 1)\,E \quad \text{(Eq-2)}$$

Where E, the penalty for inserting one or more gaps, is a value that is not just a constant. Rather the value of E can change and its value depends on the particular nucleotide (i.e., DNA/RNA) or residue (i.e., Amino Acid) that is being replaced with a gap.

Yet, another version involves using the above gap penalty but the score is computed slightly differently as follows

$$S_{i,j} = max \begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ \left[S_{i-n_{gap1},j} + g(n_{gap1})\right]_{1 \leq n_{gapi} \leq i} \\ \left[S_{i,j-n_{gap2}} + g(n_{gap2})\right]_{1 \leq n_{gap2} \leq j} \end{cases}$$

There are even more variations of the above as well. Each having their own purpose, strengths and limitations. However, for the purposes of this course and example, we will use the version defined in Eq-1.

$$S_{i,j} = max \begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \text{ for gaps that run horizontally} \\ S_{i,j-1} + g(n_{gap}) \text{ for gaps that run veritally} \end{cases}$$

where $g(n_{gap})$ is defined above, and cur pos to R means from the current position in the matrix toward the right and cur pos to T means from the current position toward top.

Suppose we would like to align the following two sequences

H E A G
P Y A E

← *Similar to but not exactly like the sequences you need to align for your in-class exercise.*

In addition, suppose we have an initial gap penalty $I = 2$ and $E = 3$. Use the BLOSUM62 matrix to compute the Needleman-Wunsch global alignment score for the above sequences. Use the value of I to initialize the penalty column and row in the matrix below.

H E A G
P Y A E

This is cell location $i=1, j= 1$

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1				
2	Y	-2				
3	A	-3				
4	E	-4				
	j col →	*0*	*1*	*2*	*3*	*4*

Initial penalty row

Initial penalty column

1. Computing the score for location $i,j = 1,1$

i row ⇩			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2			
2	Y	-2				
3	A	-3				
4	E	-4				
	j col ⇨	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{0,0} + s(H,P) \\ S_{0,1} + g(n_{gap}) \\ S_{1,0} + g(n_{gap}) \end{cases} = max\begin{cases} 0 + -2 \\ -1 + -1 - 1 * 2 \\ -1 + -1 - 1 * 2 \end{cases}$$

$$= max\begin{cases} -2 \\ -4 \\ -4 \end{cases} = \textbf{-2}$$

The circled number 1 represents moving one column left
The circled number 1 represents moving one row up.

2. Computing the score for location $i,j = 1,2$

i row ⇩			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2		
2	Y	-2				
3	A	-3				
4	E	-4				
	j col ⇨	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{0,1} + s(E,P) \\ S_{0,2} + g(n_{gap}) \\ S_{1,1} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -1 + -1 \\ -2 + \max(-1 - 1 * 2, \; -1 - 2 * 2) \\ -2 + -1 - 1 * 2 \end{cases} = max\begin{cases} -2 \\ -2 + -3 = -5 \\ -2 + -3 = -5 \end{cases} = \textbf{-2}$$

NOTE: The reason for is that we now need to consider the gap values for two columns to the left of cell location $i,j = 0,2$, and the procedure for this version of NW requires us to use the *max* of the gap scores toward the left. The reason for not needing to take the *max* for cell location $i,j = 1,1$ is because we are already 1 row away from the top row.

3. Computing the score for location $i,j = 1, 3$

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	**-3**	
2	Y	-2				
3	A	-3				
4	E	-4				
	j col →	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{0,2} + s(A,P) \\ S_{0,3} + g(n_{gap}) \\ S_{1,2} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -2 + -1 \\ -3 + \max(-1 - 1 * 2, \; -1 - 2 * 2, \; -1 - 3 * 2) \\ -2 + -1 - 1 * 2 \end{cases} = max\begin{cases} -3 \\ -3 + -3 = -6 \\ -2 + -3 = -5 \end{cases} =$$

-3

4. Computing the score for location $i,j = 1, 4$

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	**-5**
2	Y	-2				
3	A	-3				
4	E	-4				
	j col →	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{0,3} + s(G,P) \\ S_{0,4} + g(n_{gap}) \\ S_{1,3} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -3 + -2 \\ -4 + \max(-1 - 1 * 2, \dots, \dots, \; -1 - 4 * 2) \\ -3 + -1 - 1 * 2 \end{cases} = max\begin{cases} -5 \\ -4 + -3 = -7 \\ -3 + -3 = -6 \end{cases} = \textbf{-5}$$

5. Computing the score for location $i,j = 2, 1$

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	**1**			
3	A	-3				
4	E	-4				
	j col →	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{1,0} + s(H,Y) \\ S_{1,1} + g(n_{gap}) \\ S_{2,0} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -1+2 \\ -2 + -1 - 1*2 \\ -2 + \max(-1-1*2, \dots, -1-2*2) \end{cases} = max\begin{cases} 1 \\ -2 + -3 = -5 \\ -2 + -3 = -5 \end{cases} = \mathbf{1}$$

6. Computing the score for location $i,j = 2, 2$

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2		
3	A	-3				
4	E	-4				
	j col →	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{1,1} + s(E,Y) \\ S_{1,2} + g(n_{gap}) \\ S_{2,1} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -2+-2 \\ -2 + \max(\dots, -1-1*2) \\ 1 + \max(\dots, -1-1*2) \end{cases} = max\begin{cases} -4 \\ -2 + -3 = -5 \\ 1 + -3 = -2 \end{cases} = \mathbf{-2}$$

7. Computing the score for location $i,j = 2, 3$

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2	-4	
3	A	-3				
4	E	-4				
	j col →	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{1,2} + s(A,Y) \\ S_{1,3} + g(n_{gap}) \\ S_{2,2} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -2 + -2 \\ -3 + \max(\dots, -1 - 1 * 2) \\ -4 + \max(\dots, -1 - 1 * 2) \end{cases} = max\begin{cases} -4 \\ -3 + -3 = -6 \\ -4 + -3 = -7 \end{cases} = \mathbf{-4}$$

8. Computing the score for location $i,j = 2, 4$

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2	-4	-6
3	A	-3				
4	E	-4				
	j col →	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{1,3} + s(G,Y) \\ S_{1,4} + g(n_{gap}) \\ S_{2,3} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -3 + -3 \\ -5 + \max(\dots, -1 - 1 * 2) \\ -4 + \max(\dots, -1 - 1 * 2) \end{cases} = max\begin{cases} -6 \\ -5 + -3 = -8 \\ -4 + -3 = -7 \end{cases} = \mathbf{-6}$$

9. Computing the score for location $i,j = 3, 1$

i row ⬇			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2	-4	-6
3	A	-3	-2			
4	E	-4				
	j col ➡	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{2,0} + s(H, A) \\ S_{2,1} + g(n_{gap}) \\ S_{3,0} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -2 + -2 \\ 1 + \max(\dots, -1 - 1 * 2) \\ -3 + \max(\dots, -1 - 1 * 2) \end{cases} = max\begin{cases} -4 \\ 1 + -3 = -2 \\ -3 + -3 = -6 \end{cases} = \textbf{-2}$$

10. Computing the score for location $i,j = 3, 2$

i row ⬇			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2	-4	-6
3	A	-3	-2	0		
4	E	-4				
	j col ➡	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{2,1} + s(E, A) \\ S_{2,2} + g(n_{gap}) \\ S_{3,1} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} 1 + -1 \\ -2 + \max(\dots, -1 - 1 * 2) \\ -2 + \max(\dots, -1 - 1 * 2) \end{cases} = max\begin{cases} 0 \\ -2 + -3 = -5 \\ -2 + -3 = -5 \end{cases} = \textbf{0}$$

11. Computing the score for location $i,j = 3, 3$

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2	-4	-6
3	A	-3	-2	0	2	
4	E	-4				
	j col →	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{2,2} + s(A,A) \\ S_{2,3} + g(n_{gap}) \\ S_{3,2} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -2+4 \\ -4 + \max(\ldots, -1-1*2) \\ -1 + \max(\ldots, -1-1*2) \end{cases} = max\begin{cases} 2 \\ -4 + -3 = -7 \\ -1 + -3 = -4 \end{cases} = \mathbf{2}$$

12. Computing the score for location $i,j = 3, 4$

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2	-4	-6
3	A	-3	-2	0	2	-1
4	E	-4				
	j col →	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{2,3} + s(G,A) \\ S_{2,4} + g(n_{gap}) \\ S_{3,3} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -4+0 \\ -6 + \max(\ldots, -1-1*2) \\ 2 + \max(\ldots, -1-1*2) \end{cases} = max\begin{cases} -4 \\ -6 + -3 = -9 \\ 2 + -3 = -1 \end{cases} = \mathbf{-1}$$

13. Computing the score for location $i,j = 4, 1$

i row ⬇			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2	-4	-6
3	A	-3	-2	0	2	-1
4	E	-4	-3			
	j col ➡	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{3,0} + s(H,E) \\ S_{3,1} + g(n_{gap}) \\ S_{4,0} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -3+0 \\ -2 + \max(\ldots, -1-1*2) \\ -4 + \max(\ldots, -1-1*2) \end{cases} = max\begin{cases} -3 \\ -2 + -3 = -5 \\ -4 + -3 = -7 \end{cases} = \mathbf{-3}$$

14. Computing the score for location $i,j = 4, 2$

i row ⬇			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2	-4	-6
3	A	-3	-2	0	2	-1
4	E	-4	-3	3		
	j col ➡	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max\begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max\begin{cases} S_{3,1} + s(E,E) \\ S_{3,2} + g(n_{gap}) \\ S_{4,1} + g(n_{gap}) \end{cases} =$$

$$max\begin{cases} -2+5 \\ -1 + \max(\ldots, -1-1*2) \\ -3 + \max(\ldots, -1-1*2) \end{cases} = max\begin{cases} 3 \\ -1 + -3 = -4 \\ -3 + -3 = -6 \end{cases} = \mathbf{3}$$

15. Computing the score for location $i,j = 4, 3$

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2	-4	-6
3	A	-3	-2	0	2	-1
4	E	-4	-3	3	0	
	j col →	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max \begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max \begin{cases} S_{3,2} + s(A,E) \\ S_{3,3} + g(n_{gap}) \\ S_{4,2} + g(n_{gap}) \end{cases} =$$

$$max \begin{cases} -1 + -1 \\ 2 + \max(\dots, -1 - 1 * 2) \\ 3 + \max(\dots, -1 - 1 * 2) \end{cases} = max \begin{cases} -2 \\ 2 + -3 = -1 \\ 3 + -3 = 0 \end{cases} = \mathbf{0}$$

16. Computing the score for location $i,j = 4, 4$

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2	-4	-6
3	A	-3	-2	0	2	-1
4	E	-4	-3	3	0	0
	j col →	*0*	*1*	*2*	*3*	*4*

$$S_{i,j} = max \begin{cases} S_{i-1,j-1} + s(x_i, y_j) \\ S_{i-1,j} + g(n_{gap}) \\ S_{i,j-1} + g(n_{gap}) \end{cases} = max \begin{cases} S_{3,3} + s(G,E) \\ S_{3,4} + g(n_{gap}) \\ S_{4,3} + g(n_{gap}) \end{cases} =$$

$$max \begin{cases} 2 + -2 \\ -1 + \max(\dots, -1 - 1 * 2) \\ 0 + \max(\dots, -1 - 1 * 2) \end{cases} = max \begin{cases} 0 \\ -1 + -3 = -4 \\ 0 + -3 = -3 \end{cases} = \mathbf{0}$$

Starting at the lower right cell $i,j = 4,4$ and performing a traceback, the optimal alignment path is highlighted in yellow.

i row ↓			H	E	A	G
0		0	-1	-2	-3	-4
1	P	-1	-2	-2	-3	-5
2	Y	-2	1	-2	-4	-6
3	A	-3	-2	0	2	-1
4	E	-4	-3	3	0	0
	j col →	*0*	*1*	*2*	*3*	*4*

This results in the following alignment and score.

```
 _ H E  A  G
   |  |  |  |
0  1  0  2  0
   |  |  |  |
 P Y  A  _  E
```

= 0 + 2 + 0 + 1 + -1 + 0 = 2

The score, using the BLOSUM62 matrix, of the alignment before applying the NW algorithm is

```
 H E  A  G
 |  |  |  |
-2 -2  4 -2
 |  |  |  |
 P Y  A  E
```

= -2 + -2 + 4 + -2 = -2

So how does one know what version of any alignment algorithm to use. It depends on what you expect or need to find. If you need to take into account which nucleotide is replaced with a gap (e.g., if you need to consider particular nucleotide/residue groups are involved) then a version of NW mentioned in the NOTE #1 box at the top of this document might be more appropriate. If, on the other hand, you need to align two sequences with minimal gap insertions, then perhaps the original NW algorithm with a fixed and relatively high gap penalty is warranted. If you need to align a subset of two sequences, then perhaps the BLAST or Smith-Waterman local alignment algorithm is more appropriate.

REFERENCES

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.

DONE