RUTGERS, THE STATE UNIVERSITY OF NEW JERSEY

DATA STRUCTURES & ALGORITHMS TERM PAPER

Sequence Alignment Methods in Bioinformatics for Biological Sequences

---

Pratik Mistry
April 7, 2020

Table of Contents

Abstract **-** This paper will discuss a sequence alignment method in bioinformatics used for arranging the sequences of DNA, RNA, or protein i.e. biological sequences to identify regions of similarity that may be a consequence of functional, structural, or evolutionary relationships between the sequences. The Needleman-Wunch method is a sequence alignment method used to find the global alignment between two biological sequences. We will conclude that the impact of this method to find biologically good and accurate alignments can have significant meaning, showing relationships and homology between different sequences, and can provide useful information, which can be used to further identify new members of protein families.
Keywords **-** Sequence Alignment; Global Alignment; Bioinformatics

1. Introduction

The sequencing of the genomes from several organisms, and high-throughput X Ray structure analysis, have brought to the scientific community a large amount of data about the sequences and structures of several thousand proteins. This information can effectively be used for medical and biological research only if one can extract functional insight from it and bioinformatics is the computational approach that can be used to achieve this goal. Bioinformatics is an interdisciplinary field that develops methods and software tools for understanding biological data, in particular when the data sets are large and complex [1].

There are many algorithms and approaches used to study sequence and structure alignments, secondary structure prediction, functional classification of proteins, threading and modeling of distantly-related homologous proteins to modeling the progress of protein expression through a cell's life cycle. Needleman-Wunch algorithm[2] is the application of dynamic programming to find the optimal global alignments between two biological sequences.

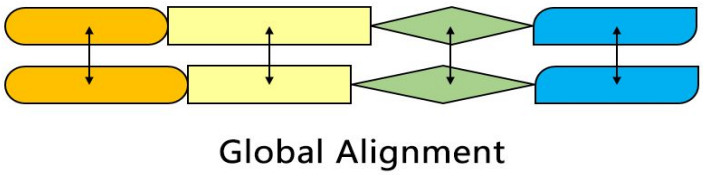

Figure 1. Global Alignment Representation *(Best viewed in color)*

From Figure 1, we can see that global alignment attempts to align every residue in every sequence, are most useful when the sequences in the query set are similar and of roughly equal size.

General steps to find optimal global alignment:

1. Determine the substitution matrix and scoring schemes - Match, Mismatch and Indel/Gap scores
2. Constructing and initializing the scoring matrix
3. Fill in the scoring matrix with scores in each cell
4. Traceback to find the global alignments
5. Choosing optimal global alignment

Optimal global alignment is based on the highest alignment score calculated between two globally aligned sequences. Thus, when a new sequence is found, the structure and function can be easily predicted by doing sequence alignment. Since it is believed that, a sequence sharing common ancestor would exhibit similar structure or function. Greater the sequence similarity, greater is the chance that they share similar structure or function.

2. Terminology

1. ***Biological Sequence***: A biological sequence is a single, continuous molecule of nucleic acid or protein. Example: G A A T T C

2. ***Substitution Matrix***: A grid that provides scores for the substitution of every amino acid (or nucleotide) for every other. Example: PAM[6] and BLOSUM[7] are widely referred substitution matrices [4]. Figure 2 shows the representation of the respective matrices

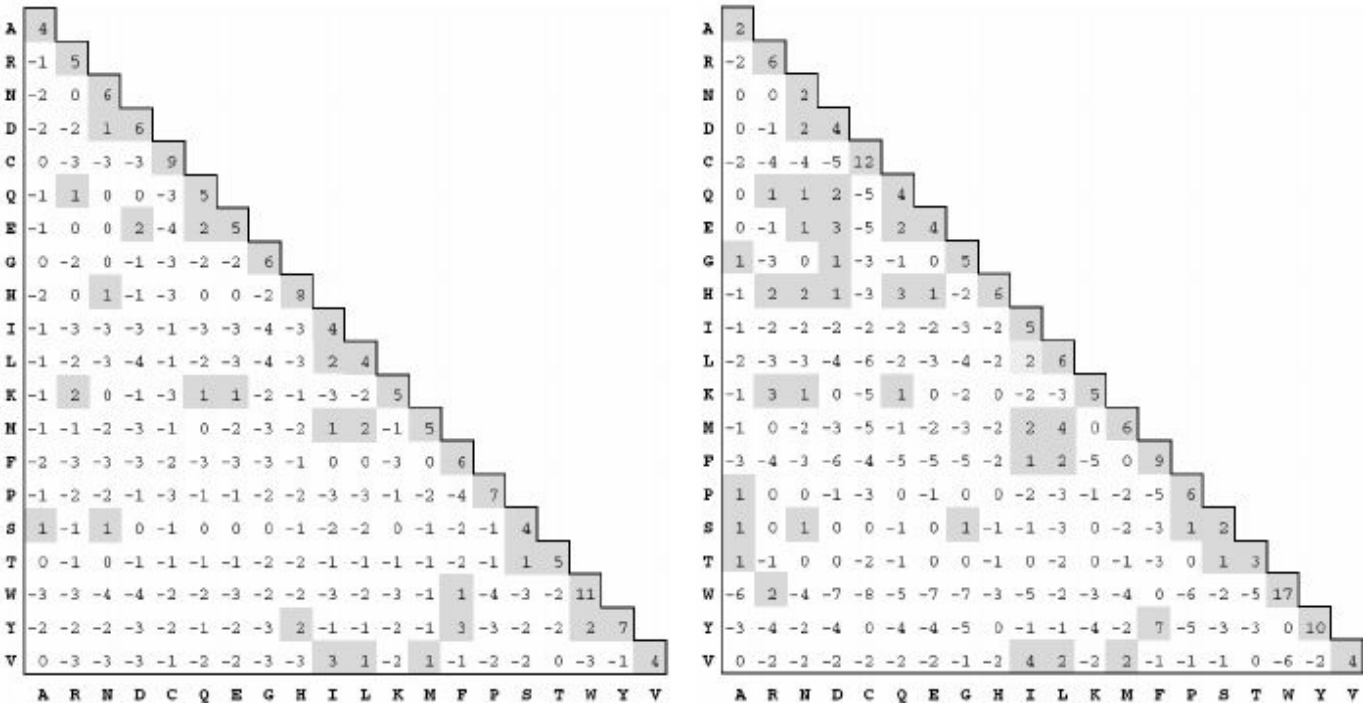

Figure 2. PAM-250 and BLOSUM-62 matrices [5]

3. ***Scoring Schemes***: A scoring scheme/system helps score each individual pair of letters. Example: Scores for Match between letters can be +1,Mismatch and Gap/Indel can be -1

4. ***Scoring Matrix:*** Scoring Matrix is used to determine the relative score made by matching two characters in a sequence alignment. Figure 3 shows the representation of simple scoring matrix between two sequences

	A	C	G	T
A	1	-1	-1	-1
C	-1	1	-1	-1
G	-1	-1	1	-1
T	-1	-1	-1	1

Figure 3. Simple Scoring Matrix with Match = 1 and Mismatch = -1

5. ***Gaps and Gap Penalty:*** Gaps i.e. indels are gaps that may be created between proteins (letters) after finding optimal alignment. Gap penalty is the score that should be considered for each gap while calculating the score of an alignment.

6. ***Traceback:*** Traceback is the method of finding the alignment sequences between two sequences from a scoring matrix. It starts with the rightmost bottom cell of the scoring matrix and ends at the origin i.e. leftmost top cell.

7. ***Global Sequence Alignment***: The best alignment over the entire length of two sequences when the two sequences are of similar length, with a significant degree of similarity throughout. Example:

```
S  I  M  I  L  A  R  I  T  Y
|  |  |  |  |  |  |  |  |  |
P  I  -  L  L  A  R  -  -  -
```

3. Needleman-Wunsch Method

The Needleman–Wunsch algorithm[2] is one of the applications of dynamic programming used to align and compare biological sequences. The algorithm essentially divides a large problem (e.g. the full sequence) into a series of smaller problems, and it uses the solutions to the smaller problems to find an optimal solution to the larger problem. It is also sometimes referred to as the *optimal matching algorithm* and the *global alignment technique*.

The Needleman–Wunsch algorithm[2] is still widely used for optimal global alignment, particularly when the quality of the global alignment is of the utmost importance. The algorithm assigns a score to every possible alignment, and the purpose of the algorithm is to find all possible alignments having the highest score. Basically, a good alignment has minimum gaps and mismatches inserted into the sequences, but at the same time maximize the number of positions in the aligned strings that match.

3.1. Methodology

In this section, we will discuss the detailed steps of the algorithm in-order to find optimal aligned sequence between two small DNA sequences represented as strings:

Sequence A = SEND

Sequence B = AND

and since lengths of both sequences approximately the same i.e. 4 and 3, global alignment can be easily founded using the defined algorithm.

3.1.1. Determining substitution matrix and Scoring Schemes

This is a very critical step which makes a huge difference in results of alignment.

A. Substitution Matrix

- As defined in earlier sections, we can either choose PAM (**Point Accepted Mutation)** [6] or BLOSUM (**BLO**cks **SU**bstitution **M**atrix) [7] predefined matrices. It helps us determine the score between pairs of protein elements.
- In this paper, we won't use any of these matrices to calculate score between elements. Instead we will consider the generalised scoring scheme mentioned below.

B. Scoring Schemes

- It helps us in calculating the score between all the pairs of the elements in the sequence. Below is one kind of scoring scheme:
  Match = +1
  Mismatch = -1
  Indel (INsertion or DELetion) or Gap = -1
- *Match*: Occurs if two letters at the current index are the same
- *Mismatch:* Occurs if two letters at the current index are different
- *Indel or Gap:* Occurs if best alignment involves one letter aligning to a gap in the other string
- Scoring schemes helps us in filling scoring matrix and also calculate scores of alignments after traceback to help us choose best sequence alignment

3.1.2. Constructing and Initializing the scoring matrix

In this step, we construct and initialize the scoring matrix *Fi,j*.

A. Constructing the Scoring Matrix

- The scoring matrix created is of size (m+1,n+1) where m and n are lengths of sequence strings A and B. Figure 4 represents the scoring matrix *Fi,j*

		A	N	D
	F(0,0)	*F(0,1)*	*F(0,2)*	*F(0,3)*
S	*F(1,0)*	*F(1,1)*	*F(1,2)*	*F(1,3)*
E	*F(2,0)*	*F(2,1)*	*F(2,2)*	*F(2,3)*
N	*F(3,0)*	*F(3,1)*	*F(3,2)*	*F(3,3)*
D	*F(4,0)*	*F(4,1)*	*F(4,2)*	*F(3,3)*

Figure 4. Constructed Scoring Matrix *Fi,j*

B. Initializing the Scoring Matrix

- The first cell value is always 0 which is considered origin
- Given there are no 'top' or 'top-left' cells for the first row only the existing cell to the left can be used to calculate the score of each cell. Hence a gap penalty (-1 in this case) is added for each shift to the right as this represents an indel from the previous score. This results in the first row being 0, −1, −2, −3.
- The same applies to the first column as only the existing score above each cell can be used as seen in the Figure 5
- This initialization completely depends on the *gap penalty or Indel value*
- Figure 5 represents the initialized scoring matrix.

		A	N	D
	0 →	-1 →	-2 →	-3
S	-1	*F(1,1)*	*F(1,2)*	*F(1,3)*
E	-2	*F(2,1)*	*F(2,2)*	*F(2,3)*
N	-3	*F(3,1)*	*F(3,2)*	*F(3,3)*
D	-4	*F(4,1)*	*F(4,2)*	*F(4,3)*

Figure 5. Initialized Scoring Matrix

3.1.3. Fill in the scoring matrix

- In this step, we calculate scores of each cell of the matrix starting with cell F(1,1).
- Figure 6 depicts the pictorial representation of scoring cell F(i,j)

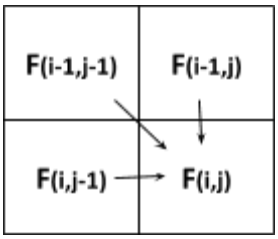

Figure 6. Pictorial Representation of Scoring a Cell

- Score of any cell F(i,j) is represented by formula mentioned in equation (1)

$$F(i,j) = \max \begin{cases} F(i-1,j-1) + S(A_i,B_j) \\ F(i-1,j) + g \\ F(i,j-1) + g \end{cases} \quad (1)$$

where $0 <= i <= m$ and $0 <= j <= n$

Equation 1: Scoring Formula of a cell

Where, g is the gap penalty

F(i-1,j-1) is score of top left diagonal cell

F(i-1,j) is score of upper cell

F(i,j-1) is score of left cell

S(*Ai,Bj*) is the match or mismatch score between elements at *Ai* and *Bj*.

Note: Generally value of S(*Ai,Bj*) *is referred from substitution matrix[4] like PAM[6] or BLOSUM[7] in real world*

- For e.g: Since Match = 1, Mismatch = -1 and Gap = -1, F(1,1) is calculated as shown in equation (2) and the equivalent partial scoring matrix is shown in Figure 7:

$$F(1,1) = \max \begin{cases} F(i-1,j-1) + S(A_i,B_j) = 0-1 = -1 \\ F(i-1,j) + g = -1-1 = -2 \\ F(i,j-1) + g = -1-1 = -2 \end{cases} = -1 \quad (2)$$

Equation 2: Scoring formula for cell F(1,1)

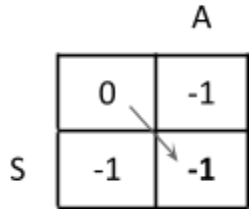

Figure 7. Partial scoring matrix

- Similarly, we calculate scores of all the cells of the scoring matrix and the resulting scoring matrix is represented in Figure 8.
  NOTE: The arrows indicate from which cell (left, top or diagonal) the value of the current cell can be determined.

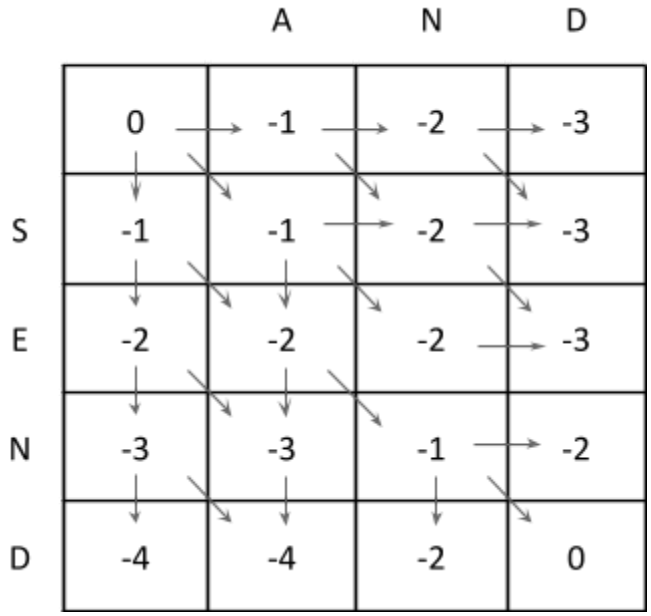

Figure 8. Resultant Scoring Matrix

3.1.4. Traceback to find global alignments

- In this step, we find all the global alignments by traceback method where we start from the last cell of matrix and trace to origin i.e. from F(4,3) to F(0,0)
- Below are the set of rules that we need to consider while constructing sequences [2]:
  - A diagonal arrow represents a match or mismatch, so the letter of the column and the letter of the row of the origin cell will align
  - A horizontal or vertical arrow represents an indel. Horizontal arrows will align a gap ("-") to the letter of the row (the "side" sequence), vertical arrows will align a gap to the letter of the column (the "top" sequence)

- If there are multiple arrows to choose from, they represent a ***branching*** of the alignments. If two or more branches all belong to paths from the bottom right to the top left cell, they are equally viable alignments.

- The scoring/traceback matrix can be represented as seen in Figure 9.
  NOTE: The arrows indicate from which cell (left, top or diagonal) the value of the current cell can be determined. Arrows are reversed as we perform traceback

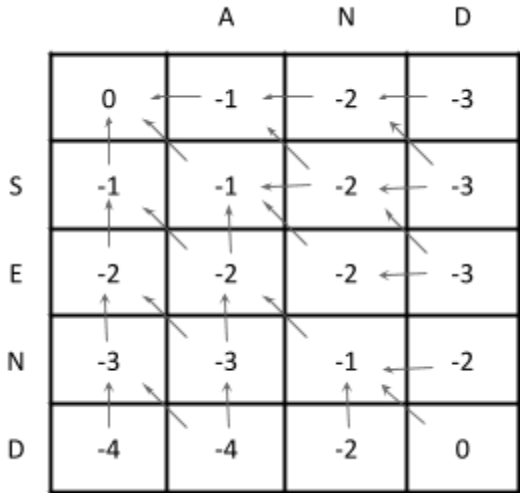

Figure 9. Possible traceback paths for finding sequences

- Based on the rules and scoring matrix the aligned sequences by ***exhaustive method*** of traceback are:
  Sequence A: D → ND → END → SEND
  Sequence B: D → ND → AND → - AND
  ↓ *(branching)*
  → END → SEND
  → - ND → A-ND

3.1.5. Choosing optimal global alignment

- Once we get all the global aligned sequences between A and B, we find the scores of each alignment and find optimal i.e. best global alignment based on scoring schemes
- For the first alignment:
  Score = Match * 2 + Mismatch * 1 + Gap * 1 = (1)(2) + (-1)(1) + (-1)(1) = 0
- Similarly, for the second alignment: Score = 0
- Since the maximum score is 0, we can choose either of them as the optimal global alignment sequence.
- In reality, the exhaustive method complexity in terms of compute and time increases with larger protein sequences as there would be many branching and resulting alignments

- Needleman-Wunch algorithm[2] uses dynamic programming technique and directly gives us the optimal global alignment in very less time instead of finding all the sequences.

3.1.6. Pseudo-code for the algorithm

- Scoring Matrix, F*i,j* will be assigned to be the optimal score for the alignment of the first i=0,...,m characters in A and the first j=0,...,n characters in B. The principle of optimality is then applied as follows:
- Basis:

  *F(i,0) = g * i*

  *F(0,j) = g * j*
- Recursion, based on the *principle of optimality*:

  F(i,j) = max (F(i-1,j-1) + S(Ai,Bj), F(i-1,j) + g, F(i,j-1) + g)
- Algorithm 1 is the Pseudocode[2] for initializing and filling the scoring matrix *Fi,j where* A and B are the protein sequences and S is the substitution matrix[4].

Algorithm 1:

```
g ← Gap Penalty
for i=0 to length(A)
    F(i,0) ← d*i
for j=0 to length(B)
    F(0,j) ← d*j
for i=1 to length(A)
    for j=1 to length(B) {
        F(i,j) ← max(F(i−1,j−1) + S(Ai, Bj), F(i−1, j) + g,  F(i, j−1) + g)
     }
```

- Once the scoring matrix F is computed, there would be an alignment sequence whose score would be maximum.
- To compute an alignment that actually gives this score, we traceback from from the bottom right cell, and compare the value with the three possible cells (Top-left Diagonal, Left, and Top) to see which it is derived from as depicted with arrows in Figure 9 and move till the origin i.e. top-left element 0. Below are alignment scenarios[3]:
  - If derived from Top Left Diagonal cell, then *Ai* and *Bj* are aligned,
  - If from the top cell, then Bj is aligned with a gap, and
  - If from the left cell, then Ai is aligned with a gap

- In reality, more than one choice may have the same value, leading to alternative optimal alignments.
- Algorithm 2 is the Pseudocode[2] for finding the optimal global alignment.

Algorithm 2:

```
AlignmentA ← ""
AlignmentB ← ""
i ← length(A)
j ← length(B)
while (i > 0 or j > 0) {
 if (i > 0 and j > 0 and F(i,j) == F(i−1,j−1) + S(Ai, Bj)) {
  AlignmentA ← Ai + AlignmentA
  AlignmentB ← Bj + AlignmentB
  i ← i − 1
  j ← j − 1
 }
 else if (i > 0 and F(i,j) == F(i−1,j) + d) {
  AlignmentA ← Ai + AlignmentA
  AlignmentB ← "−" + AlignmentB
  i ← i − 1
 }
 else {
  AlignmentA ← "−" + AlignmentA
  AlignmentB ← Bj + AlignmentB
  j ← j − 1
 }
}
```

3.2. Analysis

- Computing the score Fi,j for each cell in the table is an **O(1)** operation. Thus the Time Complexity of the algorithm for two sequences of length m and n is **O(mn)**.
- Since the algorithm fills an m*n table the Space Complexity is **O(mn)**.
- Complexity for computing optimal alignment using Needleman-Wunch algorithm[2] is far better than Brute-Force/Exhaustive Search method as the later method calculates all global alignments and we select alignment with the highest score.
- The number of possible global alignments between two sequences of length N in *Exhaustive search* method [3] is:

$$2^{2N} / \sqrt{\pi N}$$

- For example two sequences of 250 residues, it is ~ 10^{149} which is a NP hard problem.
- The Needleman-Wunsch algorithm[2] requires filling a 250 × 250 matrix and returns the optimal global alignment in lesser time complexity all because of the technique of dynamic programming.
- The Needleman-Wunsch algorithm[2] works regardless of the length or complexity of sequences, and *guarantees* to find the best alignment.

4. Application Usage

Since there are many applications of Needleman-Wunch algorithm[2] for sequence alignment, this paper discusses its use in bioinformatics to align protein or nucleotide sequences. The alignment sequences help us in study of genomics, compare biological sequences, find relationships and homology between different sequences, and can provide useful information, which can be used to further identify new members of protein families.

Other applications would be in the field of Computer Vision for Stereo Matching which processes 3D reconstruction from a pair of stereo images[2]. It also has profound applications in the field of Process Mining in order to perform Workflow Analysis and Knowledge Discovery using Multiple Sequence Alignment[9] in business and marketing research and also healthcare. NLP is another field where this algorithm is used to partially automate the comparative method by which linguists traditionally reconstruct languages[13].

5. Related Work

Smith-Waterman algorithm[8] is another sequence alignment method that performs *local sequence alignment* which is necessary for determining similar regions between two strings of nucleic acid sequences or protein sequences. Levenshtein distance algorithm[10] is a string metric algorithm for measuring the difference between two sequences. It has also been shown that it is possible to improve the running time to O(mn/log n) using the Method of Four Russians technique[11] - a variation in Needleman-Wunch algorithm[2]. Also, related work was done in Process-oriented Iterative Multiple Alignment for Medical Process Mining [12] used mainly for workflow analysis, medical traces/process alignment and knowledge discovery.

6. Conclusion

In this paper, we discussed the detailed steps with analysis of Needleman-Wunch algorithm for biological sequence alignment used widely in the field of Bioinformatics for study of genes,protein sequences and DNA/RNA. The Needleman-Wunsch algorithm is appropriate for finding the best alignment of two sequences which are (i) of the similar length; (ii) similar across their entire lengths. This work has a very great impact in life sciences as it helps in finding the newer sequences or helps us find the homology relationships with the existing protein sequences by finding optimal global alignment sequences. Also, we discussed the wide applications of algorithms in other business domains making it very popular.

7. References

1. https://en.wikipedia.org/wiki/Bioinformatics
2. https://en.wikipedia.org/wiki/Needleman%E2%80%93Wunsch_algorithm
3. https://www.cs.sjsu.edu/~aid/cs152/NeedlemanWunsch.pdf
4. https://en.wikipedia.org/wiki/Substitution_matrix
5. https://www.researchgate.net/figure/PAM-250-and-Blosum-62-matrices_fig1_265241022
6. https://en.wikipedia.org/wiki/Point_accepted_mutation
7. https://en.wikipedia.org/wiki/BLOSUM
8. https://en.wikipedia.org/wiki/Smith%E2%80%93Waterman_algorithm#Algorithm
9. https://en.wikipedia.org/wiki/Multiple_sequence_alignment
10. https://en.wikipedia.org/wiki/Levenshtein_distance
11. https://en.wikipedia.org/wiki/Method_of_Four_Russians
12. “Process-oriented Iterative Multiple Alignment for Medical Process Mining”, S Chen, S Yang, M Zhou, R Burd, I Marsic, 2017 IEEE International Conference on Data Mining Workshops (ICDMW), 438-445
13. https://en.wikipedia.org/wiki/Sequence_alignment