Hindawi
International Journal of Mathematics and Mathematical Sciences
Volume 2020, Article ID 3858057, 9 pages
https://doi.org/10.1155/2020/3858057

*Research Article*

# Formulation and Analysis of Patterns in a Score Matrix for Global Sequence Alignment

**James Owusu Asare,[1] Justice Kwame Appati ,[2] and Kwaku Darkwah[1]**

[1]*Department of Mathematics, Kwame Nkrumah University of Science and Technology, Kumasi, Ghana*
[2]*Department of Computer Science, University of Ghana, Legon, Ghana*

Correspondence should be addressed to Justice Kwame Appati; jkappati@ug.edu.gh

Received 15 April 2020; Accepted 16 May 2020; Published 1 June 2020

Academic Editor: Aloys Krieg

Global sequence alignment is one of the most basic pairwise sequence alignment procedures used in molecular biology to understand the similarity that arises among the structure, function, or evolutionary relationship between two nucleotide sequences. The general algorithm associated with global sequence alignment is the dynamic programming algorithm of Needleman and Wunsch. In this paper, patterns are exploited in the score matrix of the Needleman–Wunsch algorithm. With the help of some examples, the general patterns realized are formulated as new a priori propositions and corollaries that are established for both equal and unequal length comparisons of any two arbitrary sequences.

## 1. Introduction

Sequence alignment is the matching of strings or sequences of characters to identify patterns that may lead to informed structural or functional relationships between the strings or sequences matched. Varying situational problems seem to require the use of sequence alignment and examples abound from computer science to molecular biology, where sequences are usually aligned to make more meaning out of them. Bare [1] has discussed how researchers have over the years considered genetic sequences as strings of characters instead of focusing on their intrinsic properties to be able to compute the similarity among two or more related sequences. Global and local sequence alignment are used for matching two sequences, and they are categorized as pairwise sequence alignment or multiple sequence alignment when any alignment matches more than two sequences [2–4]. The Needleman–Wunsch algorithm [5–7] and the Smith–Waterman algorithm [8] are the general algorithms associated with pairwise global and local alignment, respectively. These two algorithms utilize a procedure called the dynamic programming approach. Dynamic programming as coined by Bellman in the 1940s is simply the process of solving a bigger problem by finding optimal solutions to its smaller nested problems [9–11]. Thus, to tackle a problem in the context of dynamic programming, it must possess the notion of recurrence. In [12, 13], an algorithm is said to be a well-defined computational task that accepts input and produces output values after following through a systematic method. Mathematical theory is thus a prerequisite behind the designing of functional programs [14, 15], and the algorithm design specializes in solving such problems. Global sequence alignment is mentioned as one of the vast dynamic programming applications in practical problems [16–18].

More recently, Ouzounis and Baichoo [19] have stated that even though pairwise sequence alignment has been dealt with over time, concerns still remain in resolving exact evolutionary distances that demand very specific estimates. They have suggested the existence of theoretical relationships within alignments, algorithms, and data that are yet to be found. Motivated by the display of position-dependent arrays for affine gaps using the Needleman–Wunsch algorithm in [20, 21], we consider the more basic linear gap penalties using arbitrary sequences and proceed to find patterns in the score matrix which were absent in the presentation. This approach was taken because it is found missing in the available stream of literature although it can be likened to the edit graph illustration seen in [1]. To the best of our knowledge, there has never been any theoretical exposition of this kind for the most basic and very fundamental concept of constant linear gap penalties which constitute an implicit part of affine gap penalties. Thus, we focus on finding patterns in global sequence alignment using constant linear gap penalties and attempt the display of completed score matrices of the Needleman–Wunsch algorithm with distinct positional arrays that can be inspected. We give confirmation of this by basic proofs and suggest how predictions can be made. To offer the reader the much needed convenience, concepts that are deemed useful are recalled in brief with corresponding references of comprehensive works that give more in-depth information.

## 2. Needleman–Wunsch Algorithm

Let the recursive formulation for the Needleman–Wunsch algorithm be

(1) *Initialization*

Let $P(0,0) = 0$,
$P_m(i,0) = P_m(i-1,0) + \text{gap penalty}, \forall i = 1,\dots,\mu_1$ and
$P_m(0,j) = P_m(0,j-1) + \text{gap penalty}, \forall j = 1,\dots,\mu_2$, where $P(0,0)$ is the initialization pivot for the score matrix, $P_m(i,0)$ is the initial row pivot for the score matrix, and $P_m(0,j)$ is the initial column pivot for the score matrix.

(2) *Cell Box Calculation*

Let

$$P_m(i,j) = \max\begin{cases} P_D(i,j) = P_m(i-1,j-1) + \sigma(\alpha(i),\beta(j)), \\ P_R(i,j) = P_m(i-1,j) + \text{gap penalty}, \\ P_L(i,j) = P_m(i,j-1) + \text{gap penalty}, \end{cases} \tag{1}$$

where $P_m(i,j)$ is the pivot for each cell box calculation, $P_D(i,j) = P_m(i-1,j-1) + \sigma(\alpha(i),\beta(j))$ is the diagonal value of a cell box, $P_R(i,j) = P_m(i-1,j) + \text{gap penalty}$ is the right value of a cell box, $P_L(i,j) = P_m(i,j-1) + \text{gap penalty}$ is the left value of a cell box, and $\sigma(\alpha(i),\beta(j))$ is the score for aligning the sequence characters of $\alpha(i)$ and $\beta(j)$.

Specifically, the procedure for the Needleman–Wunsch algorithm follows the dynamic programming approach of the score matrix, traceback, and alignment as outlined in the subsequent sections.

*2.1. Score Matrix.* The score matrix is a tabular box constructed to keep count of score results. The score matrix for the Needleman–Wunsch algorithm begins with an initialization process and ends with the calculation of cell boxes.

*2.1.1. Initialization.* A sequence matrix is created with $\mu_1 + 1$ columns and $\mu_2 + 1$ rows in order for the initial matrix gap to be aligned where $\mu_1$ and $\mu_2$ are the lengths of arbitrary sequences, $\langle\alpha\rangle$ and $\langle\beta\rangle$, respectively. The letters of the sequence $\alpha$ fill in the horizontal axis, and similarly, the characters of $\langle\beta\rangle$ fill in the vertical axis of the sequence matrix created. Before the scoring begins from the upper left corner of the initialized matrix to the lower right corner of the matrix, the value $P(0,0) = 0$ is assigned to the intersection of the first row and the first column of the matrix (i.e., the initial gap). The reason for the gap penalty for an alignment is because of the possibility of mutation which may insert or delete a string character from one of the sequences. Arrows that point in the direction of positional movement (diagonal and left or right) are placed in each cell box of the matrix and only terminate when all the cell boxes are completely filled.

*2.1.2. Calculation of Cell Boxes in the Score Matrix*

(1) Fill the initialized gap values first on both the horizontal axis and the vertical axis with a defined constant gap value score
(2) Follow with the calculation of each cell box having the three position-dependent arrays (left/beside, right/bottom, or diagonal)
(3) Allow only a match/mismatch value for a “diagonal” position, and allow the “bottom” or “beside” positions to take linear gap values only
(4) For each computed cell box values, find the maximum score and let that be the pivot
(5) The pivot of a computed cell box directly affects the next cell boxes in the row or the column

*2.2. Traceback.* A simpler score matrix table that contains only the pivot of each cell box calculation is constructed from the original position-dependent arrays of the score matrix table. Arrow pointers are used to direct a path from the highest score or an optimal value in the matrix (which actually occurs at the lower right end corner of the matrix) and traced back to the next biggest value of the predecessors until we reach the intersection of the first row and the column with the initial gap.

*2.3. Alignment*

*2.3.1. Alignment Generation from a Traceback.* To write the sequence characters that appear from the optimal alignment path of the traceback stage, the following steps are followed:

(1) When the arrow is diagonal, write both characters in the alignment
(2) When the arrow is vertical, write the corresponding horizontal character, and in place of the vertical character, leave a gap
(3) When the arrow is horizontal, write the corresponding vertical character, and in place of the horizontal character, leave a gap

Thus, for both the vertical and horizontal arrow positions, one character and a gap are written for the alignment, where the gap explicitly replaces a character position in the alignment. An alignment can only be inferred as the best if the optimal value from the score matrix table corresponds to the alignment score calculation (based on the scoring scheme defined).

*2.3.2. The Problem of Aligning Any Two Sequences.* The problem of aligning any two sequences can be simplified as discovering the optimal means of aligning any two arbitrary sequences say $\langle\alpha\rangle$ and $\langle\beta\rangle$ such that the character "–" noted as a gap is filled into both $\langle\alpha\rangle$ and $\langle\beta\rangle$ or either of them where

(1) Any single character in $\langle\alpha\rangle$ matches a single character in $\langle\beta\rangle$ or a gap

(2) The final sum of scores from the scoring function over the aligned pairs and the gap penalties as given by the function of gap penalty is maximized

*2.3.3. Letter Choice for Arbitrary Sequences.* The letter choice of this study shall be that of DNA considered as a string of four characters of adenine—A, guanine—G, cytosine—C, and thymine—T.

## 3. Scheme for Scoring

*Definition 1.* The constant gap penalty, $v(k) = g \times k$, where $g$ is an assigned constant and $k$ is the sequence length count of $i = 1, \ldots, \mu_1$ for the score matrix row and $j = 1, \ldots, \mu_2$ for the score matrix column. This is the penalty awarded to gaps and is also known as the linear gap function.

*Definition 2.* The affine gap penalty is the penalty awarded to gaps where a greatest consecutive sequence of $k$ gaps is given as $v(k) = q + gk$, where $q$ is the penalty charged for opening the gap and $g$ is the penalty charged for extending it.

Thus, more formally we define

(1) A constant linear gap as

$$v(k) = gk. \tag{2}$$

(2) An affine gap as

$$v(k) = \begin{cases} q + gk, & k \geq 1, \\ 0, & k = 0. \end{cases} \tag{3}$$

Altschul's theory: assign positive scores to an identity and conserved replacements, and assign negative scores to less likely replacements.

*Remark 1.* Needleman and Wunsch use the "identity matrix" in scoring with 1 for a match and 0 for a mismatch. Needleman–Wunsch's score was criticised for not reflecting observations from the nature because purine-purine or pyrimidine-pyrimidine is less prone to be mutated in comparison with mutations of purine-pyrimidine. Because there is no definite defined score for DNA alignment, the choice of scoring for this study is +5 for a match, −1 for a mismatch, and −2 for a gap as proposed in [18] following the above theory.

More formally, let

$$\sigma(\alpha(i), \beta(j)) = \begin{cases} +5, & \text{for } \alpha(i), \beta(j) \text{ character match}, \\ -1, & \text{for } \alpha(i), \beta(j) \text{ character mismatch}, \end{cases} \tag{4}$$

$\forall i = 1, \ldots, \mu_1$ and $\forall j = 1, \ldots, \mu_2$.

Also, let the linear gap penalty, $v(k) = gk$, where $g$ is a constant of "−2" and $k$ is the sequence length count of $i = 1, \ldots, \mu_1$ for the score matrix row and $j = 1, \ldots, \mu_2$ for the score matrix column.

## 4. Results and Discussion

*4.1. Pattern Investigations in the Score Matrix of Needleman–Wunsch Algorithm.* The sequences $\alpha$ and $\beta$ that will be used are arbitrary sample sequences that were chosen based on consideration of the following:

(1) $\mu_1 = \mu_2$ (equal character length of sequences)

(2) $\mu_1 \neq \mu_2$ (unequal character length of sequences)

*4.1.1. Equal Character Length* $(\mu_1 = \mu_2 = \mu)$

*Example 1.* Let $\langle\alpha\rangle$ = CTTGA and $\langle\beta\rangle$ = CTAGA. Because the length of the sequence characters of $\langle\alpha\rangle$ and $\langle\beta\rangle$ is $\mu_1 = \mu_2 = 5$, respectively, we align $\alpha$ and $\beta$ in a score matrix following the above formulation.

(1) Initialization: $P(0,0) = 0$ for score matrix initialization, and then for the initial row pivot, $P_m(i,0) = P_m(i-1,0)$ + gap penalty, where $\forall k = i$, we find the gap penalty as follows:

$$\begin{aligned}
\text{for } i = 1, \quad & P_m(1,0) = P(0,0) + \text{gap penalty}, \\
& \Longrightarrow 0 + (-2) = -2, \\
\text{for } i = 2, \quad & P_m(2,0) = P_m(1,0) + \text{gap penalty}, \\
& \Longrightarrow -2 + (-2) = -4, \\
\text{for } i = 3, \quad & P_m(3,0) = P_m(2,0) + \text{gap penalty}, \\
& \Longrightarrow -4 + (-2) = -6, \\
\text{for } i = 4, \quad & P_m(4,0) = P_m(3,0) + \text{gap penalty}, \\
& \Longrightarrow -6 + (-2) = -8, \\
\text{for } i = 5, \quad & P_m(5,0) = P_m(4,0) + \text{gap penalty}, \\
& \Longrightarrow -8 + (-2) = -10,
\end{aligned} \tag{5}$$

and for the initial column pivot, $P_m(0,j) = P_m(0,j-1)$ + gap penalty, $\forall k = j$, the gap penalty is obtained as follows:

$$\begin{aligned}
&\text{for } j = 1, \quad P_m(0,1) = P(0,0) + \text{gap penalty},\\
&\qquad\qquad \Longrightarrow 0 + (-2) = -2,\\
&\text{for } j = 2, \quad P_m(0,2) = P_m(0,1) + \text{gap penalty},\\
&\qquad\qquad \Longrightarrow -2 + (-2) = -4,\\
&\text{for } j = 3, \quad P_m(0,3) = P_m(0,2) + \text{gap penalty},\\
&\qquad\qquad \Longrightarrow -4 + (-2) = -6,\\
&\text{for } j = 4, \quad P_m(0,4) = P_m(0,3) + \text{gap penalty},\\
&\qquad\qquad \Longrightarrow -6 + (-2) = -8,\\
&\text{for } j = 5, \quad P_m(0,5) = P_m(0,4) + \text{gap penalty},\\
&\qquad\qquad \Longrightarrow -8 + (-2) = -10.
\end{aligned} \tag{6}$$

Before following with further calculation, some identification is placed on each cell box for easy noting as shown in Table 1.

(2) Cell box calculations:

$$P_m(1,1) = \max\begin{cases} P_D(1,1) = P(0,0) + \sigma(C,C) = 0 + 5 = 5,\\ P_R(1,1) = P_m(0,1) + \text{gap} = -2 + (-2) = -4,\\ P_L(1,1) = P_m(1,0) + \text{gap} = -2 + (-2) = -4. \end{cases} \tag{7}$$

*Remark 2.* $\sigma(C,C)$ in $P_D(1,1)$ was "5" because that is the assigned score for matched characters. $P_m(1,1) = \max\{P_D(1,1), P_R(1,1), P_L(1,1)\} = \max[5 \ -4 \ -4] = 5$.

$$P_m(1,2) = \max\begin{cases} P_D(1,2) = P_m(0,1) + \sigma(C,T) = -2 + (-1) = -3,\\ P_R(1,2) = P_m(0,2) + \text{gap} = -4 + (-2) = -6,\\ P_L(1,2) = P_m(1,1) + \text{gap} = 5 + (-2) = 3. \end{cases} \tag{8}$$

*Remark 3.* $\sigma(C,T)$ in $P_D(1,2)$ was "$-1$" because that is the assigned score for mismatched characters. $P_m(1,2) = \max\{P_D(1,2), P_R(1,2), P_L(1,2)\} = \max[-3, -6, 3] = 3$.

$$P_m(1,3) = \max\begin{cases} P_D(1,3) = P_m(0,2) + \sigma(C,A) = -4 + (-1) = -5,\\ P_R(1,3) = P_m(0,3) + \text{gap} = -6 + (-2) = -8,\\ P_L(1,3) = P_m(1,2) + \text{gap} = 3 + (-2) = 1. \end{cases} \tag{9}$$

*Remark 4.* $\sigma(C,A)$ in $P_D(1,3)$ was "$-1$" because that is the assigned score for mismatched characters. $P_m(1,3) = \max\{P_D(1,3), P_R(1,3), P_L(1,3)\} = \max[-5, \ -8, \ 1] = 1$.

Consequently, the recursion follows similarly until all the cell boxes are filled. In the event of a pivot tie in a cell box, one tie value is picked. For the tabular display, these notations are used interchangeably; $P_D(i,j)$ is the same as DV, $P_L(i,j)$ is the same as LV, $P_R(i,j)$ is the same as RV, and $P_m(i,j)$ is the same as the pivot.

Table 1: Identifying cell boxes of CTTGA and CTAGA.

| | | C | T | T | G | A |
|---|---|---|---|---|---|---|
| | 0 | −2 | −4 | −6 | −8 | −10 |
| C | −2 | i | ii | iii | iv | v |
| T | −4 | vi | vii | viii | ix | x |
| A | −6 | xi | xii | xiii | xiv | xv |
| G | −8 | xvi | xvii | xviii | xix | xx |
| A | −10 | xxi | xxii | xxiii | xxiv | xxv |

*4.1.2. Pattern Results for Equal Character Length.* With reference to Tables 1 and 2 and Figure 1,

(1) The filled-in cell boxes for column 1, i.e., cell boxes (i, vi, xi, xvi, and xxi), have values coinciding with that of row 1, .i.e., cell boxes (i, ii, iii, iv, and v)
(2) The leading value (pivot) of each cell box in both row 1 and column 1 remains the same
(3) The bottom value in column 1 switches to become the beside value in row 1 and vice versa
(4) For each 3 pointed arrow intersecting any 4 cell boxes, the beside value of a 2nd cell box also coincides with the bottom value of a 3rd cell box
(5) For each cell box, the value of the preceding bottom value is less than the value of the immediate next adjacent diagonal value

*4.1.3. Traceback and Alignment for Equal Character Length.* The traceback and alignment stages for equal character length of sequences are, respectively, shown in this section.

Figure 2 shows the pivot of each cell box calculation of the score matrix table of CTTGA and CTAGA in Figure 1. Thus, Figure 2 is a simpler score matrix table constructed from the original score matrix table of CTTGA and CTAGA in Figure 1. The arrow pointers direct the path from the optimal value and traceback to the initialization value of zero. Based on the traceback and the diagonal direction of the arrow pointers, the alignment is written as

$$\begin{gathered}\text{CTTGA},\\ \text{CTAGA}.\end{gathered} \tag{10}$$

To confirm the correctness of the alignment done, we check using calculations. Recall the scoring scheme: match = +5, mismatch = −1, and gap = −2. We have four alignment matches: $C - C$, $T - T$, $G - G$, and $A - A$ and one mismatched alignment: $T - A$. Hence,

$$5 + 5 - 1 + 5 + 5 = 19, \tag{11}$$

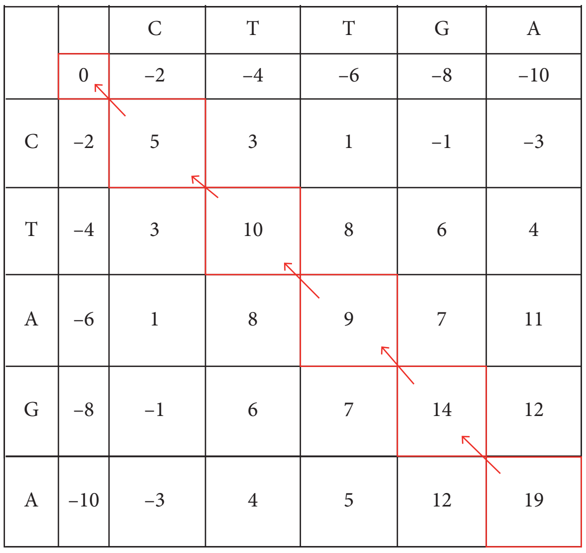
which is the same as the optimal value from the score matrix table. The alignment is thus optimal.

*4.1.4. Unequal Character Length* $(\mu_1 \neq \mu_2)$

*Example 2.* Suppose $\langle\alpha\rangle$ = AGCTG and $\langle\beta\rangle$ TCAG, then to fill in the score matrix values of the cell boxes, the prior-stated recursive formulation is used. This results in Figures 3 and 4.

TABLE 2: Filled-in table of CTTGA and CTAGA.

| | | | | |
|---|---|---|---|---|
| i. | DV = 5 | LV = −4 | RV = −4 | Pivot = 5 |
| ii. | DV = −3 | LV = 3 | RV = −6 | Pivot = 3 |
| iii. | DV = −5 | LV = 1 | RV = −8 | Pivot = 1 |
| iv. | DV = −7 | LV = −1 | RV = −10 | Pivot = −1 |
| v. | DV = −9 | LV = −3 | RV = −12 | Pivot = −3 |
| vi. | DV = −3 | LV = −6 | RV = 3 | Pivot = 3 |
| vii. | DV = 10 | LV = 1 | RV = 1 | Pivot = 10 |
| viii. | DV = 8 | LV = 8 | RV = −1 | Pivot = 8 |
| ix. | DV = 0 | LV = 6 | RV = −3 | Pivot = 6 |
| x. | DV = −2 | LV = 4 | RV = −5 | Pivot = 4 |
| xi. | DV = −5 | LV = −8 | RV = 1 | Pivot = 1 |
| xii. | DV = 2 | LV = −1 | RV = 8 | Pivot = 8 |
| xiii. | DV = 9 | LV = 6 | RV = 6 | Pivot = 9 |
| xiv. | DV = 7 | LV = 7 | RV = 4 | Pivot = 7 |
| xv. | DV = 11 | LV = 5 | RV = 2 | Pivot = 11 |
| xvi. | DV = −7 | LV = −10 | RV = −1 | Pivot = −1 |
| xvii. | DV = 0 | LV = −3 | RV = 6 | Pivot = 6 |
| xviii. | DV = 7 | LV = 4 | RV = 7 | Pivot = 7 |
| xix. | DV = 14 | LV = 5 | RV = 5 | Pivot = 14 |
| xx. | DV = 6 | LV = 12 | RV = 9 | Pivot = 12 |
| xxi. | DV = −9 | LV = −12 | RV = −3 | Pivot = −3 |
| xxii. | DV = −2 | LV = −5 | RV = 4 | Pivot = 4 |
| xxiii. | DV = 5 | LV = 2 | RV = 5 | Pivot = 5 |
| xxiv. | DV = 6 | LV = 3 | RV = 12 | Pivot = 12 |
| xxv. | DV = 19 | LV = 10 | RV = 10 | Pivot = 19 |



| | | C | T | T | G | A |
|---|---|---|---|---|---|---|
| | 0 | −2 | −4 | −6 | −8 | −10 |
| C | −2 | 5 | 3 | 1 | −1 | −3 |
| T | −4 | 3 | 10 | 8 | 6 | 4 |
| A | −6 | 1 | 8 | 9 | 7 | 11 |
| G | −8 | −1 | 6 | 7 | 14 | 12 |
| A | −10 | −3 | 4 | 5 | 12 | 19 |

FIGURE 2: Traceback in the score matrix of CTTGA and CTAGA.

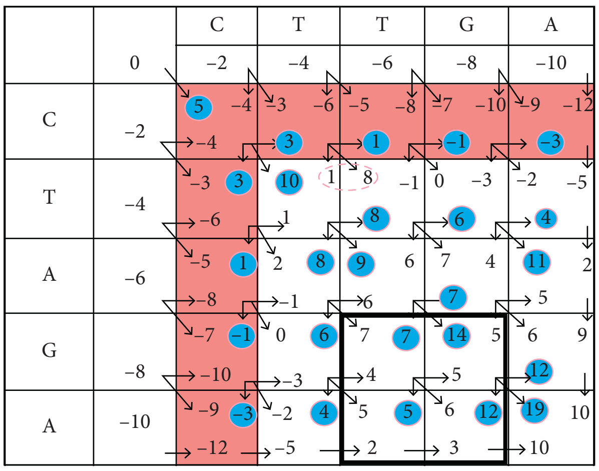


FIGURE 1: Score matrix of the cell boxes of CTTGA and CTAGA.

### 4.1.5. *Pattern Results for Unequal Character Length*

(1) The filled-in cell boxes for column 1, i.e., (i, vi, xi, and xvi) cell boxes, have consistent values with that of row 1, i.e., (i, ii, iii, and iv), up until where they terminate at (v) and at the inconsistent (iii)

(2) Except for where an inconsistency is noted at (v) and (iii), the pivot values for each cell box of row 1 and column 1 remain the same

(3) For each 3 pointed arrows intersecting any 4 cell boxes, the beside value of a 2nd cell box also coincides with the bottom value of a 3rd cell box

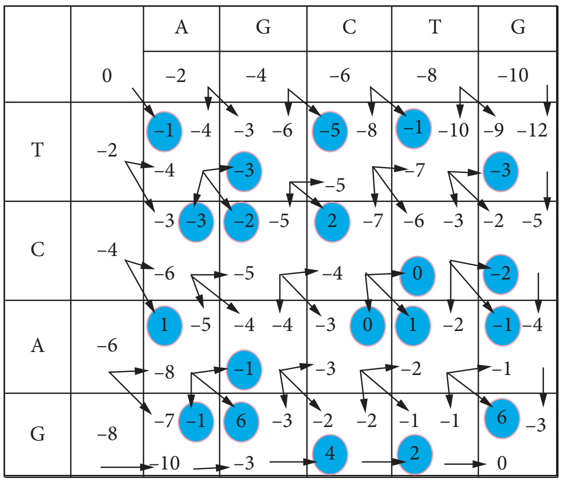


FIGURE 3: Score matrix of AGCTG and TCAG.

| | | A | G | C | T | G |
|---|---|---|---|---|---|---|
| | 0 | −2 | −4 | −6 | −8 | −10 |
| T | −2 | *i* | *ii* | *iii* | *iv* | *v* |
| C | −4 | *vi* | *vii* | *vii* | *ix* | *x* |
| A | −6 | *x* | *xii* | *xiii* | *xiv* | *xv* |
| G | −8 | *xvi* | *xvii* | *xviii* | *xix* | *xx* |

FIGURE 4: Labelling of the cell boxes of AGCTG and TCAG.

(4) For each cell box, the value of the preceding bottom value is seen to be always less than the value of the immediate next adjacent diagonal value

*4.1.6. Traceback and Alignment for Unequal Character Length.* The traceback and alignment stages for the example on unequal character length of sequences are, respectively, shown below. Figure 5 shows the pivot of each cell box calculation of the score matrix table of AGCTG and TCAG of Table 3. Based on the traceback and the direction of the arrow pointers, the alignment is written as

$$\begin{gathered}\text{AGCTG,}\\ \text{T}-\text{CAG.}\end{gathered} \tag{12}$$

We check the correctness of the alignment done using calculations. Recall the scoring scheme: match = +5, mismatch = −1, and gap = −2. We have two alignment matches: $G-G$ and $C-C$, two alignment mismatches: $AT$ and $TA$, and one gapped alignment: $G\ -$ . Hence,

$$-1-2+5-1+5=6, \tag{13}$$

which is the same as the optimal value from the score matrix table. The alignment is thus optimal.

*4.2. Propositions and Proofs for Equal Sequence Length.* The following propositions are *a priori* results deduced from the pattern results of equal character length of sequences.

*Definition 3.* Let the linear gap penalty be $v(k) = gk$, where $g$ is a constant of −2 and $k$ is the sequence length count of $k = 1,\ldots,\mu_1$ for the score matrix row and $k = 1,\ldots,\mu_2$ for the score matrix column.

*Definition 4.* Define $P(0,0) = 0$ to be the initialization pivot for the score matrix.

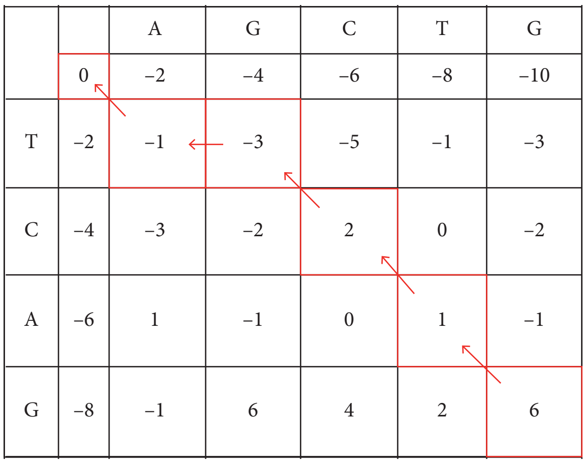
*Definition 5.* Define $P_m(i,0) = P_m(i-1,0)$ + gap penalty, $\forall i = 1,\ldots,\mu_1$, to be the initial row pivot for the score matrix.

*Definition 6.* Define $P_m(0,j) = P_m(0,j-1)$ + gap penalty, $\forall j = 1,\ldots,\mu_2$, to be the initial column pivot for the score matrix.

*Definition 7.* Define

$$P_m(i,j) = \max\begin{cases} P_D(i,j) = P_m(i-1,j-1) + \sigma(\alpha(i),\beta(j)),\\ P_R(i,j) = P_m(i-1,j) + \text{gap penalty},\\ P_L(i,j) = P_m(i,j-1) + \text{gap penalty},\end{cases} \tag{14}$$

$\forall i = 1,\ldots,\mu_1$ and $\forall j = 1,\ldots,\mu_2$, where $P_m(i,j)$ is the pivot for each cell box calculation, $P_D(i,j) = P_m(i-1,j-1) + \sigma(\alpha(i),\beta(j))$ is the diagonal value of a cell box, $P_R(i,j) = P_m(i-1,j)+$ gap penalty is the right value of a cell box, $P_L(i,j) = P_m(i,j-1)+$ gap penalty is the left value of a cell box, and $\sigma(\alpha(i),\beta(j))$ is the score for aligning the sequence characters of $\alpha(i)$ and $\beta(j)$.



| | | A | G | C | T | G |
|---|---|---|---|---|---|---|
| | 0 | −2 | −4 | −6 | −8 | −10 |
| T | −2 | −1 | −3 | −5 | −1 | −3 |
| C | −4 | −3 | −2 | 2 | 0 | −2 |
| A | −6 | 1 | −1 | 0 | 1 | −1 |
| G | −8 | −1 | 6 | 4 | 2 | 6 |

Figure 5: Traceback in the score matrix of AGCTG and TCAG.

Table 3: Filled-in table of AGCTG and TCAG.

| | | | | |
|---|---|---|---|---|
| i. | DV = −1 | LV = −4 | RV = −4 | Pivot = −1 |
| ii. | DV = −3 | LV = −3 | RV = −6 | Pivot = −3 |
| iii. | DV = −5 | LV = −5 | RV = −8 | Pivot = −5 |
| iv. | DV = −1 | LV = −7 | RV = −10 | Pivot = −1 |
| v. | DV = −9 | LV = −3 | RV = −12 | Pivot = −3 |
| vi. | DV = −3 | LV = −6 | RV = −3 | Pivot = −3 |
| vii. | DV = −2 | LV = −5 | RV = −5 | Pivot = −2 |
| viii. | DV = 2 | LV = −4 | RV = −7 | Pivot = 2 |
| ix. | DV = −6 | LV = 0 | RV = −3 | Pivot = 0 |
| x. | DV = −2 | LV = −2 | RV = −5 | Pivot = −2 |
| xi. | DV = 1 | LV = −8 | RV = −5 | Pivot = 1 |
| xii. | DV = −4 | LV = −1 | RV = −4 | Pivot = −1 |
| xiii. | DV = −3 | LV = −3 | RV = 0 | Pivot = 0 |
| xiv. | DV = 1 | LV = −2 | RV = −2 | Pivot = −2 |
| xv. | DV = −1 | LV = −1 | RV = −4 | Pivot = −1 |
| xvi. | DV = −7 | LV = −10 | RV = −1 | Pivot = −1 |
| xvii. | DV = 6 | LV = −3 | RV = −3 | Pivot = 6 |
| xviii. | DV = −2 | LV = 4 | RV = −2 | Pivot = 4 |
| xix. | DV = −1 | LV = 2 | RV = −1 | Pivot = 2 |
| xx. | DV = 6 | LV = 0 | RV = −3 | Pivot = 6 |

*Definition 8.* Define

$$\sigma(\alpha(i),\beta(j)) = \begin{cases} +5, & \text{for } \alpha(i),\beta(j) \text{ character match},\\ -1, & \text{for } \alpha(i),\beta(j) \text{ character mismatch},\end{cases} \tag{15}$$

$\forall i = 1,\ldots,\mu_1$ and $\forall j = 1,\ldots,\mu_2$.

**Proposition 2.** *Filled-in cell boxes for $P_m(1,j), \forall j = 1,\ldots,\mu_2$ are analogous to filled-in cell boxes for $P_m(i,1), \forall i = 1,\ldots,\mu_1$ for equal length comparison of sequences.*

*Proof.* By Definition 7, we have

$$\begin{gathered} P_m(1,j) = \max\begin{cases} P_D(1,j) = P_m(0,j-1) + \sigma(\alpha(1),\beta(j)),\\ P_R(1,j) = P_m(0,j) + \text{gap},\\ P_L(1,j) = P_m(1,j-1) + \text{gap},\end{cases}\\ P_m(i,1) = \max\begin{cases} P_D(i,1) = P_m(i-1,0) + \sigma(\alpha(i),\beta(1)),\\ P_R(i,1) = P_m(i-1,1) + \text{gap},\\ P_L(i,1) = P_m(i,0) + \text{gap},\end{cases}\end{gathered} \tag{16}$$

$\forall i, j = 1, \ldots, \mu$ since $\mu_1 = \mu_2 = \mu$. Again, by Definitions 5 and 6,

$$\begin{aligned} P_m(i,0) &= P_m(i-1,0) + \text{gap}, \\ &\Longrightarrow P_L(i,1) = P_m(i,0) + \text{gap} \\ &= \left(P_m(i-1,0) + \text{gap}\right) + \text{gap}, \end{aligned} \tag{17}$$

also,

$$\begin{aligned} P_m(0,j) &= P_m(0,j-1) + \text{gap}, \\ &\Longrightarrow P_R(1,j) = P_m(0,j) + \text{gap} \\ &= \left(P_m(0,j-1) + \text{gap}\right) + \text{gap}. \end{aligned} \tag{18}$$

Using Definitions 5 and 6 and applying induction, $P_m(1,0) = P(0,0) + \text{gap}$ and $P_m(0,1) = P(0,0) + \text{gap}$, thus $P_m(1,0) = P_m(0,1)$ is true. Assume $P_m(k,0) = P_m(0,k)$ is always true, then

$$\begin{aligned} P_m(k+1,0) &= P_m(k,0) + \text{gap} \\ &= P_m(0,k) + \text{gap} \\ &= P_m(0,k+1), \\ \therefore P_m(i,0) &= P_m(0,j). \end{aligned} \tag{19}$$

Recall that the gap is constant, so we write $P_L(i,1) = P_m(i,0) + \text{gap} = P_R(1,j) = P_m(0,j) + \text{gap}$, thus $P_L(i,1) = P_R(1,j)$ which completes the first part of our proof.

Again, from $P_L(1,j) = P_m(1,j-1) + \text{gap}$, $P_R(i,1) = P_m(i-1,1) + \text{gap}$, and by induction, let $i = 1 = j$, then

$$\begin{aligned} P_L(1,j) &= P_m(1,0) + \text{gap} \\ &= P(0,0) + \text{gap} + \text{gap} \\ &= P(0,0) + 2\,\text{gaps}, \\ P_R(i,1) &= P_m(0,1) + \text{gap} \\ &= P(0,0) + \text{gap} + \text{gap} \\ &= P(0,0) + 2\text{gaps}. \end{aligned} \tag{20}$$

Assume $P_m(k,0) = P_m(0,k)$, then

$$\begin{aligned} P_m(k+1,0) &= P_m(k,0) + \text{gap} \\ &= P_m(0,k) + \text{gap} \\ &= P_m(0,k+1), \\ \therefore P_R(i,1) &= P_L(1,j), \quad \text{where } i = j, \end{aligned} \tag{21}$$

which completes the second part of our proof.

Recall,

$$\begin{aligned} P_L(i,1) &= P_m(i-1,0) + 2\,\text{gaps}, \\ P_R(1,j) &= P_m(0,j-1) + 2\,\text{gaps}, \\ P_L(i,1) &= P_R(1,j) \text{was established.} \end{aligned} \tag{22}$$

It follows that $P_m(i-1,0) = P_m(0,j-1)$ is obvious. Thus, $P_D(1,j) = P_m(0,j-1) + \sigma(\alpha(1),\beta(j))$ and $P_D(i,1) = P_m(i-1,0) + \sigma(\alpha(i),\beta(1))$. We are left to show that $\sigma(\alpha(1),\beta(j)) = \sigma(\alpha(i),\beta(1))$.

For $i = j = 1$, $\sigma(\alpha(1),\beta(j)) = \sigma(\alpha(i),\beta(1))$ since $\alpha(1)$ and $\beta(1)$ match. For $i = j \neq 1$, $\alpha(1)$ and $\beta(j)$ mismatch and $\alpha(i)$ and $\beta(1)$ also mismatch. Thus,

$$\sigma(\alpha(1),\beta(j)) = \sigma(\alpha(i),\beta(1)), \quad \forall i, j \in 1, \ldots, \mu. \tag{23}$$

Hence, $P_D(1,j) = P_D(i,1)$ since $P_m(i-1,0) = P_m(0,j-1)$. □

**Corollary 3.** *The right value of filled-in cell boxes for $P_m(1,j), \forall j = 1, \ldots, \mu_2$ becomes the left value of filled-in cell boxes for $P_m(i,1), \forall i = 1, \ldots, \mu_1$ and vice versa.*

*Proof.* From Proposition 2 and Definitions 5–7, it is clear that

$$\begin{aligned} P_L(i,1) &= P_R(1,j), \\ P_R(i,1) &= P_L(1,j), \\ &\text{where } i = j. \end{aligned} \tag{24}$$

□

**Corollary 4.** *The pivot values, $P_m(1,j), \forall j = 1, \ldots, \mu_2$ and $P_m(i,1), \forall i = 1, \ldots, \mu_1$, are the same for comparison of equal length of sequences.*

*Proof.* By Definition 7,

$$\begin{aligned} P_m(1,j) &= \max\{P_D(1,j), P_R(1,j), P_L(1,j)\}, \quad \forall j = 1, \ldots, \mu_2, \\ P_m(i,1) &= \max\{P_D(i,1), P_R(i,1), P_L(i,1)\}, \quad \forall i = 1, \ldots, \mu_1, \end{aligned} \tag{25}$$

and by Proposition 2, we can write that $P_m(1,j) = P_m(i,1)$. Hence, it is proved. □

**Proposition 5.** *For each three pointed arrows intersecting any four cell boxes, the left value of a 2nd cell box corresponds to the right value of a 3rd cell box.*

*Proof.* Let $P_m(i,j), P_m(i+1,j), P_m(i,j+1)$, and $P_m(i+1,j+1)$ be any four cell boxes, then we are to show that $P_L(i,j+1) = P_R(i+1,j)$. By Definition 7,

$$
\begin{gathered}
P_m(i,j) = \max \begin{cases} P_D(i,j) = P_m(i-1,j-1) + \sigma(\alpha(i),\beta(j)), \\ P_R(i,j) = P_m(i-1,j) + \text{gap penalty}, \\ P_L(i,j) = P_m(i,j-1) + \text{gap penalty}, \end{cases} \\
P_m(i+1,j) = \max \begin{cases} P_D(i+1,j) = P_m(i,j-1) + \sigma(\alpha(i+1),\beta(j)), \\ P_R(i+1,j) = P_m(i,j) + \text{gap penalty}, \\ P_L(i+1,j) = P_m(i+1,j-1) + \text{gap penalty}, \end{cases} \\
P_m(i,j+1) = \max \begin{cases} P_D(i,j+1) = P_m(i-1,j) + \sigma(\alpha(i),\beta(j+1)), \\ P_R(i,j+1) = P_m(i-1,j+1) + \text{gap penalty}, \\ P_L(i,j+1) = P_m(i,j) + \text{gap penalty}, \end{cases} \\
P_m(i+1,j+1) = \max \begin{cases} P_D(i+1,j+1) = P_m(i,j) + \sigma(\alpha(i+1),\beta(j+1)), \\ P_R(i+1,j+1) = P_m(i,j+1) + \text{gap penalty}, \\ P_L(i+1,j+1) = P_m(i+1,j) + \text{gap penalty}, \end{cases} \\
P_R(i+1,j) = P_m(i,j) + \text{gap penalty}, \\
P_L(i,j+1) = P_m(i,j) + \text{gap penalty}.
\end{gathered} \tag{26}
$$

Since the linear gap penalty is equal in both cases and it is obvious that $P_m(i,j)$ is the same in both cases, we write $P_L(i,j+1) = P_R(i+1,j)$. □

**Corollary 6.** *For any two adjacent row cell boxes, the right value of a preceding cell box is less than the diagonal value of the next cell box.*

*Proof.* Let $P_m(i,j)$ and $P_m(i,j+1)$ be any two adjacent cell boxes. Then, by Proposition 5, we write

$$
\begin{aligned}
P_R(i,j) &= P_m(i-1,j) + \text{gap penalty}, \\
P_D(i,j+1) &= P_m(i-1,j) + \sigma(\alpha(i),\beta(j+1)).
\end{aligned} \tag{27}
$$

We are left to show that gap penalty $< \sigma(\alpha(i),\beta(j+1))$. □

*Remark 5.* By Altschul's theory, a match and mismatch are chosen to be greater than a gap penalty. We recall the scoring scheme of the constant $g$ being "−2" in the linear gap penalty $gk$ and the diagonal score being "−1" when there is a mismatch and "+5" when there is a match.

For any $k$ count, $\forall k = 1,\ldots,\mu$, the linear gap penalty $gk$ decreases, and thus the gap penalty can never be greater than or equal to any of the diagonal scores allocated for match and mismatch. Thus,

$$
\begin{gathered}
\text{gap penalty } \underbrace{\sigma(\alpha(i),\beta(j+1))}_{\substack{\text{mismatch diagonal score} \\ \text{match diagonal score}}}, \\
\forall i,j = 1,\ldots,\mu.
\end{gathered} \tag{28}
$$

Hence, $P_R(i,j) < P_D(i,j+1), \forall i,j = 1,\ldots,\mu$.

*4.3. Proposition and Proof for Unequal Sequence Length.* The following proposition holds *a priori* from the pattern results of unequal character length of sequences. We state and prove the following general results by adhering to the same definitions stated earlier under the equal character length of sequences.

**Proposition 7.** *For each three pointed arrows intersecting any four cell boxes, the left value of a 2nd cell box corresponds to the right value of a 3rd cell box.*

*Proof.* Let $P_m(i,j), P_m(i+1,j), P_m(i,j+1)$, and $P_m(i+1,j+1)$ be any four cell boxes, then we are to show that $P_m(i+1,j) = P_m(i,j+1)$. Refer to the proof of Proposition 5. Despite the unequal sequence length of $\mu_1 \neq \mu_2$, the supposed disparity in length has no bearing on the proof. □

**Corollary 8.** *For any two adjacent row cell boxes, the right value of a preceding cell box is less than the diagonal value of the next cell box.*

*Proof.* Refer to the proof of Corollary 6. Again, despite the unequal sequence length of $\mu_1 \neq \mu_2$, the supposed disparity in length has no bearing on the proof. □

## 5. Conclusion

In this paper, the score matrix of the Needleman–Wunsch algorithm was exploited for possible patterns. Given any two arbitrary sequences of equal or unequal length, a general pattern was formulated as new *a priori* propositions and corollaries. These new formulated propositions and corollaries are justified with their corresponding proofs.

## Data Availability

No data were used to support this study.

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

## References

[1] J. C. Bare, *The Evolution of Sequence Comparison Algorithms*, University of Washington, Washington, DC, USA, 2005.

[2] S. A. de Carvalho, “Sequence alignment algorithms,” Master’s thesis, Kings College, London, UK, 2003.

[3] S. R. McAllister, R. Rajgaria, and C. A. Floudas, “Global pairwise sequence alignment through mixed-integer linear programming: a template-free approach,” *Optimisation Methods and Software*, vol. 22, no. 1, pp. 127–144, 2007.

[4] S. Batzoglou, “The many faces of sequence alignment,” *Briefings in Bioinformatics*, vol. 6, no. 1, pp. 6–22, 2005.

[5] S. B. Needleman and C. D. Wunsch, “A general method applicable to the search for similarities in the amino acid sequence of two proteins,” *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, 1970.

[6] V. Likic, “The needleman-wunsch algorithm for sequence alignment,” *Lecture given at the 7th Melbourne Bioinformatics Course*, pp. 1–46, Bi021 Molecular Science and Biotechnology Institute, University of Melbourne, Melbourne, Australia, 2008.

[7] T. Naveed, I. S. Siddiqui, and S. Ahmed, *Parallel Needleman—Wunsch Algorithm for Grid*, Bahria University, Islamabad, Pakistan, 2005.

[8] T. F. Smith, M. S. Waterman, and W. M. Fitch, “Comparative biosequence metrics,” *Journal of Molecular Evolution*, vol. 18, no. 1, pp. 38–46, 1981.

[9] S. R. Eddy, “What is dynamic programming?” *Nature Biotechnology*, vol. 22, no. 7, pp. 909-910, 2004.

[10] S. Dreyfus, “Richard bellman on the birth of dynamic programming,” *Operations Research*, vol. 50, no. 1, pp. 48–51, 2002.

[11] R. Bellman, *Dynamic Programming*, Princeton University Press, Long Island, NY, USA, 1957.

[12] D. W. Mount, *Sequence and Genome Analysis*, Cold Spring Harbor Laboratory Press, Cold Spring Harbor, NY, USA, 2nd edition, 2004.

[13] A. Donkor, K. Darkwah, J. Appati, P. Gakpleazi, and W. Naninja, “Optimal allocation of funds for loans using karmarkar’s algorithm: capital rural bank, sunyani-ghana,” *Journal of Economics, Management and Trade*, vol. 12, pp. 1–13, 2018.

[14] B. Boatemaa, J. K. Appati, and K. F. Darkwah, “Multi-criteria ranking of voice transmission carriers of a telecommunication company using promethee,” *Applied Informatics*, vol. 5, no. 1, 2018.

[15] K. Gyimah, J. K. Appati, K. Darkwah, and K. Ansah, “An improved geo-textural based feature extraction vector for offline signature verification,” *Journal of Advances in Mathematics and Computer Science*, vol. 32, no. 2, pp. 1–14, 2019.

[16] B. Bhowmik, “Dynamic programming-its principles applications strengths and limitations,” *International Journal of Engineering Science and Technology*, vol. 2, no. 7, 2010.

[17] X. Huang and K.-M. Chao, “A generalized global alignment algorithm,” *Bioinformatics*, vol. 19, no. 2, pp. 228–233, 2003.

[18] Amrita, *Global Alignment of Two Sequences—Needleman-Wunsch Algorithm*, 2012.

[19] C. A. Ouzounis and S. Baichoo, “Computational complexity of algorithms for sequence comparison, short-read assembly and genome alignment,” *Biosystems*, vol. 156-157, pp. 72–85, 2017.

[20] R. C. Edgar, “Optimizing substitution matrix choice and gap parameters for sequence alignment,” *Bmc Bioinformatics*, vol. 10, no. 1, p. 396, 2009.

[21] S. F. Altschul and B. W. Erickson, “Optimal sequence alignment using affine gap costs,” *Bulletin of Mathematical Biology*, vol. 48, no. 5-6, pp. 603–616, 1986.