Bulletin of Mathematical Biology (2004) **66**, 1423–1438
doi:10.1016/j.bulm.2004.01.005

Pairwise Alignment of the DNA Sequence Using Hypercomplex Number Representation

JIAN-JUN SHU* AND LI SHAN OUW
School of Mechanical and Production Engineering,
Nanyang Technological University,
50 Nanyang Avenue,
Singapore 639798,
Singapore
E-mail: mjjshu@ntu.edu.sg

A new set of DNA base-nucleic acid codes and their hypercomplex number representation have been introduced for taking the probability of each nucleotide into full account. A new scoring system has been proposed to suit the hypercomplex number representation of the DNA base-nucleic acid codes and incorporated with the method of dot matrix analysis and various algorithms of sequence alignment. The problem of DNA sequence alignment can be processed in a rather similar way to pairwise alignment of the protein sequence.

1. INTRODUCTION

Deoxyribonucleic acid, DNA, is the molecule of life. DNA is a double helix comprising two DNA strands running antiparallel to each other and is made of many units of nucleotides, which each consist of a sugar, a phosphate and a base. The four types of nucleotide (A, T, G and C) are linked in different orders in the extremely long DNA molecules, thus allowing a unique DNA sequence for each of the infinite number of living organisms.

With more DNA sequences becoming available (Lim and Shu, 2001, 2002), computer programs have been developed to analyze these sequences in various ways. The dot matrix method, which is used to detect similarities between sequences, was discovered first (Mount, 2001). In this method of comparing two sequences, a graph is drawn with one sequence written across a page from left to right and another sequence down the page on the left-hand side. A dot is placed where the corresponding nucleotide in the two sequences is the same. The graph is then scanned for diagonals of dots, which reveal similarities. Unless the sequences are known to be very much alike, the dot matrix method was used first as this method displayed any possible sequence alignments as diagonals on the matrix. The dot

*Author to whom correspondence should be addressed.

0092-8240/04/051423 + 16 $30.00/0

matrix analysis reveals the presence of insertions/deletions, and direct/inverted repeats that are more difficult to find by other methods. The major limitation of the dot matrix analysis is that most dot matrix computer programs do not show an actual alignment.

As the dot matrix method does not identify similarities that are interrupted, the method of sequence alignment was devised (Durbin *et al*., 1998). Sequence alignment is a procedure of comparing two sequences by searching for a series of individual characters that are in the same order. An alignment is generated, starting at the ends of the two sequences, by attempting to match all possible pairs of characters between the sequences, following a certain algorithm for matches, mismatches and gaps. This procedure generates a matrix of numbers that represent all possible alignments between the sequences. The optimal alignment between the two sequences is one that gives a highest score. The dynamic programming method is guaranteed in a mathematical sense to provide the optimal alignment for a given set of user-defined variables, including the choice of scoring matrix and gap penalties. There are two types of sequence alignment: global alignment (Needleman and Wunsch, 1970) and local alignment (Smith and Waterman, 1981). In global alignment, the entire sequences are aligned from beginning to end. It is better to use global alignment for aligning sequences that are similar and have approximately the same length. In local alignment, parts of the sequences with the most matches are aligned, giving rise to a number of subalignments in the aligned sequences. Thus local alignments are more suitable for aligning sequences that are similar only along some of their lengths, sequences that differ in length and sequences that share a conserved region or domain. These two methods of sequence comparison are sometimes used hand in hand, for more efficient sequence analysis of DNA. In this paper, the hypercomplex number system has been explored for its possible application in DNA sequencing.

2. DNA Base-Nucleic Acids in Hypercomplex Number Representation

By permutation and combination, the total number of possible mixed DNA base-nucleic acid codes is $2^4 = 16$. Since there are four types of nucleotide, a four-dimensional space is essential to represent the DNA codes fully. The hypercomplex number system required here is a third-order system of the form $Z = P_A + P_T\vec{i} + P_G\vec{j} + P_C\vec{k} = (P_A, P_T, P_G, P_C)$. To assign the values for P_A, P_T, P_G and P_C, the probability of each DNA base appearing in the DNA base-nucleic acid codes is taken into consideration. The values of P_A, P_T, P_G and P_C indicate the probabilities of the bases A, T, G and C respectively, satisfying the basic principle that $P_A + P_T + P_G + P_C = 1$.

Based on the principle in the previous section, the hypercomplex number representations of the DNA base-nucleic acid codes were derived and these are listed in Table 1.

Table 1. DNA base-nucleic acid codes and their hypercomplex number representation.

Symbol	Meaning	Explanation	Hypercomplex number representation
O	No base	No base	(0, 0, 0, 0)
A	A	Adenine	(1, 0, 0, 0)
T	T	Thymine	(0, 1, 0, 0)
G	G	Guanine	(0, 0, 1, 0)
C	C	Cytosine	(0, 0, 0, 1)
W	A or T	Weak interactions 2 h bonds	(1/2, 1/2, 0, 0)
R	A or G	puRine	(1/2, 0, 1/2, 0)
M	A or C	aMino	(1/2, 0, 0, 1/2)
K	G or T	Keto	(0, 1/2, 1/2, 0)
Y	C or T	pYrimidine	(0, 1/2, 0, 1/2)
S	C or G	Strong interactions 3 h bonds	(0, 0, 1/2, 1/2)
D	A, G or T not C	D follows C in alphabet	(1/3, 1/3, 1/3, 0)
H	A, C or T not G	H follows G in alphabet	(1/3, 1/3, 0, 1/3)
V	A, C or G not T	V follows U in alphabet	(1/3, 0, 1/3, 1/3)
B	C, G or T not A	B follows A in alphabet	(0, 1/3, 1/3, 1/3)
N	Any base	Any base	(1/4, 1/4, 1/4, 1/4)

3. DOT MATRIX WITH HYPERCOMPLEX NUMBER REPRESENTATION

The dot matrix sequence analysis is a method used primarily for comparing two sequences to look for possible alignment of characters between the sequences (Mount, 2001). It could also be used to find direct or inverted repeats in DNA sequences. The major advantage of the dot matrix method is that all possible matches of residues between the two sequences are found and significant ones are easily identifiable.

In the comparison of two sequences using the dot matrix method, one sequence $(X_1, X_2, \ldots, X_n)$ is listed across the top from the left to the right and the other sequence $(Y_1, Y_2, \ldots, Y_m)$ is listed on the left-hand side starting from the top. Beginning with the first symbol Y_1 in the sequence $(Y_1, Y_2, \ldots, Y_m)$, a dot is placed in the column when the symbol X_i is the same as Y_1, keeping to the first row. Then the second symbol Y_2 is compared to the entire sequence $(X_1, X_2, \ldots, X_n)$, placing a dot in the second row when there is a match between X_i and Y_2. This continues until the whole sequence $(Y_1, Y_2, \ldots, Y_m)$ is compared to $(X_1, X_2, \ldots, X_n)$.

Isolated dots throughout the matrix merely represent random matches, which are not related to any significant alignment. Such random matches might be too many, making the dot matrix too noisy for identifying aligning sequence regions easily. Filtering of the random matches to reduce the noise can be done by using a sliding window to compare the sequences. Instead of comparing each single sequence position, a window of adjacent positions in the two sequences are compared at the same time, placing a dot only if a minimal number of matches occurs in that window, meaning that a dot is placed only when the stringency condition is met. The window starts at the positions in X and Y to be compared and includes symbols

in a diagonal line going down and to the right, comparing each pair in turn, as in making an alignment.

With the many diagonals, it is difficult to identify sequence alignments by the dot matrix method. By performing a count of dots in all possible diagonal lines through the matrix to determine statistically which diagonals have the most matches, and by comparing these match scores with the results of random sequence comparisons, identification of the alignments is aided.

The dots matrix analysis is used to find direct and inverted repeats within sequences. Hence repeated regions in whole chromosomes are often detected by means of dot matrix analysis.

Sometimes a dot matrix analysis reveals the repeats of a sequence character when comparing a sequence against itself on a dot matrix; these repeats appear as horizontal or vertical rows of dots which sometimes merge into a rectangular pattern. The occurrence of such repeats of the same sequence symbol greatly increases the difficulty of aligning sequences as they create alignments with artificially high scores. Another situation that poses a similar problem occurs in low complexity regions. In such regions, only a few sequence characters are found, thus making it difficult to find alignments with other sequences.

In the dot matrix analysis using hypercomplex number representation of DNA bases, whether a dot is placed in a comparison of two DNA sequences is determined by the dot product of the hypercomplex number representation of the DNA base-nucleic acids and a truncation value set. The probability of finding a match between the sequences is implied in the dot product since the hypercomplex number representation assigned to each of the DNA base acids is based on the probability that each base appeared in Table 1. For instance, in a comparison of two sequences, an alignment between residues H and S, having the hypercomplex number representation $(\frac{1}{3}, \frac{1}{3}, 0, \frac{1}{3})$ and $(0, 0, \frac{1}{2}, \frac{1}{2})$ respectively, the dot product value is derived as $Z_H \cdot Z_S = (\frac{1}{3}, \frac{1}{3}, 0, \frac{1}{3}) \cdot (0, 0, \frac{1}{2}, \frac{1}{2}) = 0.17$. In other words, based on the dot product value (between 0 and 1) of the hypercomplex number representation of the residues in each sequence being compared, the truncation is set at the value of 1 (i.e., any value less than 1 will be truncated to 0) for the conventional dot matrix analysis (Mount, 2001). Unlike in the conventional dot matrix (Mount, 2001), it is now a choice to set the truncation value for a desired stringency in finding a possible match: a higher value for higher stringency. For example, regions of short matching alignment may not be necessary. In order to prevent short diagonals from appearing too frequently and making the matrix too noisy to identify actual required aligned regions, a higher truncation value may be selected so as to reduce the number of dots between the two sequences. To illustrate the influence of various factors on the outcome of a dot matrix diagram, the following pair of sequences is selected as an example:

T	G	R	B	W	B	H	K	M	W	C	Y
S	Y	A	G	M	W	D	S	H	V	R	K

	T	G	R	B	W	B	H	K	M	W	C	Y
S	0	0.5	0.25	0.33	0	0.33	0.17	0.25	0.25	0	0.5	0.25
Y	0.5	0	0	0.33	0.25	0.33	0.33	0.25	0.25	0.25	0.5	0.5
A	0	0	0.5	0	0.5	0	0.33	0	0.5	0.5	0	0
G	0	1	0.5	0.33	0	0.33	0	0.5	0	0	0	0
M	0	0	0.25	0.17	0.25	0.17	0.33	0	0.5	0.25	0.5	0.25
W	0.5	0	0.25	0.17	0.5	0.17	0.33	0.25	0.25	0.5	0	0.25
D	0.33	0.33	0.33	0.22	0.33	0.22	0.22	0.33	0.17	0.33	0	0.17
S	0	0.5	0.25	0.33	0	0.33	0.17	0.25	0.25	0	0.5	0.25
H	0.33	0	0.17	0.22	0.33	0.22	0.33	0.17	0.33	0.33	0.33	0.33
V	0	0.33	0.33	0.22	0.17	0.22	0.22	0.17	0.33	0.17	0.33	0.17
R	0	0.5	0.5	0.17	0.25	0.17	0.17	0.25	0.25	0.25	0	0
K	0.5	0.5	0.25	0.33	0.25	0.33	0.17	0.5	0	0.25	0	0.25

Figure 1. The dot product values of the hypercomplex number representation per aligned residue pair of the example sequences.

The varying parameters in the illustrations include the truncation value, the window size and the stringency (the minimum requirement on the number of dots to be present in the window before a dot is placed between the alignment of the residues). Using the above calculation, the dot product of the alignment between each of the residues of the example sequences is obtained and this is shown in a matrix in Fig. 1.

3.1. ***Effect of truncation value on dot matrix analysis.*** Based on the dot product value per aligned residue of the example sequences, a comparison between the dot matrix diagram was made and this is shown in Fig. 2, where the truncation values are at 0.3 and 0.5 respectively. The sequences are compared on a one-to-one residue basis. The dots are placed where the dot product values of the corresponding residues meet the designed truncation value.

Varying the truncation value changes the number of dots appearing on the dot matrix diagram. For the case of Fig. 2(a), many dots are present, as the truncation value is set relatively low. The high concentration of dots on the diagram makes it deceive one into thinking that there are many matched regions. However, after inserting diagonals, it is obvious that many dots are not collinear. They are only random matches all over the matrix. In addition, the number of aligned regions is also higher in Fig. 2(a) than in Fig. 2(b) as the truncation value indicates stringency in finding matches. With a lower truncation value in Fig. 2(a), we are actually looking for a higher number of possible matches, even with a smaller probability than a more certain alignment as in Fig. 2(b), which has a higher truncation value.

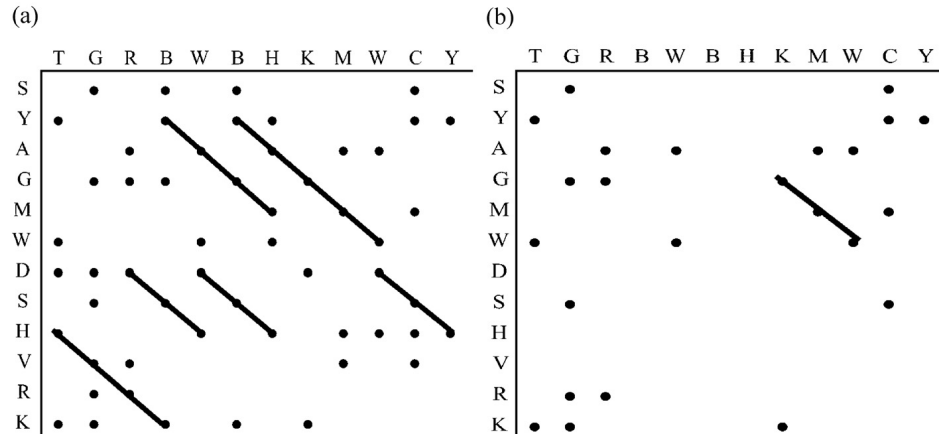

Figure 2. Dot matrices of example sequences with truncation values of (a) 0.3 and (b) 0.5.

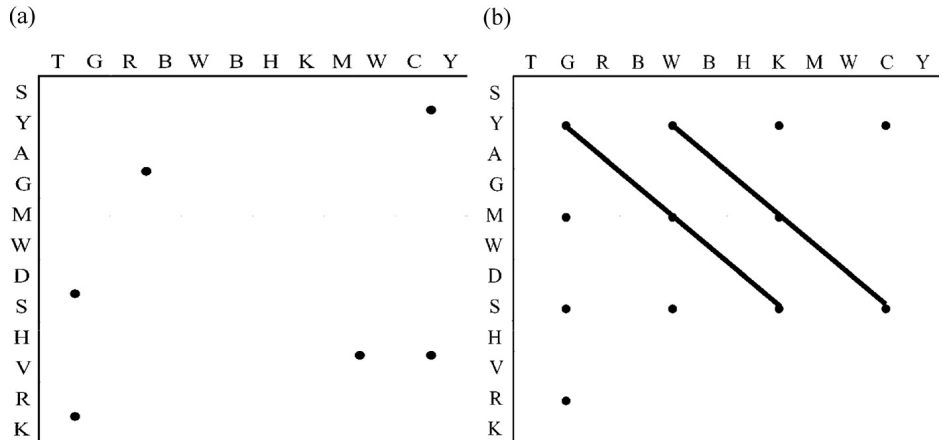

Figure 3. Dot matrices of example sequences with sliding window sizes of (a) 2 by 2 and (b) 3 by 3.

3.2. ***Effect of window size on dot matrix analysis.*** Using a truncation value of 0.3 and a stringency of 2 in each window for the dot matrix analysis for the example sequences, the influence of the sliding window size on the dot matrix diagram is investigated.

In Fig. 3(a), a window size of 2 by 2 is used. With a small window, the number of dots that can be present in each window is very small. The stringency of 3 is relatively high for a window size like that in Fig. 3(a); thus few dots are placed in the matrix. No diagonals are located with this combination of parameters on the example sequences as the dots are sparse and randomly located across the matrix.

For a larger window size, as in Fig. 3(b), the stringency of 3 now becomes a lower stringency relative to the window size. More regions meet the requirement

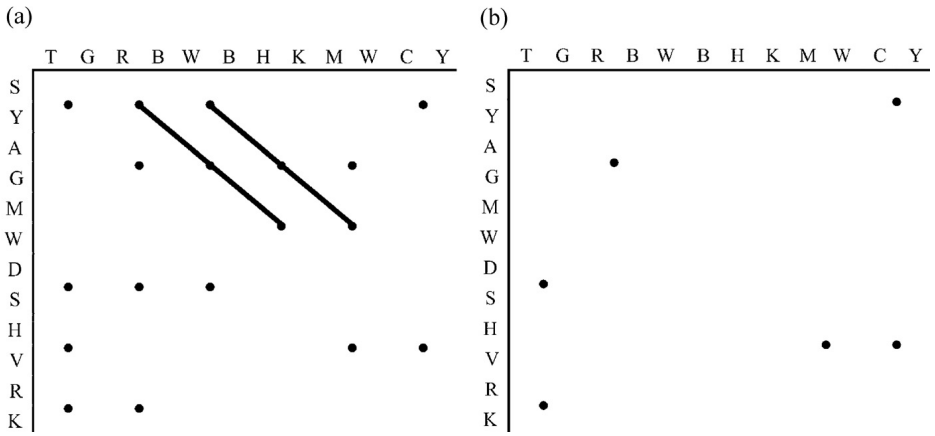

Figure 4. Dot matrices of example sequences of the same window size of 2 by 2 with stringency (a) 2 and (b) 3.

and more dots can been seen appearing even through the same pair of sequences is being used.

3.3. ***Effect of stringency on dot matrix analysis.*** As discussed earlier, the influence of the stringency desired in each dot matrix analysis will greatly determine the outcome of the dot matrix diagram. As illustrated in Fig. 4, two slightly different stringencies are used and a great difference is detected in the diagrams.

Using the same truncation value and window size, the stringency is set at 2 for Fig. 4(a) and 3 for Fig. 4(b). In Fig. 4(a), a relatively high number of dots are present with two regions of matches whereas in Fig. 4(b) the dots are so sparsely and randomly located that no matched regions can be detected.

Despite the small difference in the stringency, the two diagrams obtained are very different. This is because the level of stringency is not only determined by its value but also coupled with the window size. If the window size is larger, a slight change in the stringency will not contribute to a big difference in the dot matrix diagram. However, when the window size is very small, the difference in stringency becomes relatively important.

4. SCORING MODEL FOR HYPERCOMPLEX NUMBER REPRESENTATION

In sequence analysis by a scoring matrix, the factors to consider include the type of alignment, the scoring system used to rank alignments, the algorithm used to find optimal scoring alignments and the statistical methods used to evaluate the significance of an alignment score (Durbin *et al*., 1998). When the two sequences being compared have diverged from a common ancestor, evidence of mutation and selection could be detected. The basis mutational processes are substitutions,

	A	T	G	C	W	R	M	K	Y	S	D	H	V	B	N
	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5
A	-5	15	-5	-5	-5	5	5	5	-5	-5	-5	2	2	2	-5	0
T	-5	-5	15	-5	-5	5	-5	-5	5	5	-5	2	2	-5	2	0
G	-5	-5	-5	15	-5	-5	5	-5	5	-5	5	2	-5	2	2	0
C	-5	-5	-5	-5	15	-5	-5	5	-5	5	5	-5	2	2	2	0
W	-5	5	5	-5	-5	5	0	0	0	0	-5	2	2	-2	-2	0
R	-5	5	-5	5	-5	0	5	0	0	-5	0	2	-2	2	-2	0
M	-5	5	-5	-5	5	0	0	5	-5	0	0	-2	2	2	-2	0
K	-5	-5	5	5	-5	0	0	-5	5	0	0	2	-2	-2	2	0
Y	-5	-5	5	-5	5	0	-5	0	0	5	0	-2	2	-2	2	0
S	-5	-5	-5	5	5	-5	0	0	0	0	5	-2	-2	2	2	0
D	-5	2	2	2	-5	2	2	-2	2	-2	-2	2	-1	-1	-1	0
H	-5	2	2	-5	2	2	-2	2	-2	2	-2	-1	2	-1	-1	0
V	-5	2	-5	2	2	-2	2	2	-2	-2	2	-1	-1	2	-1	0
B	-5	-5	2	2	2	-2	-2	-2	2	2	2	-1	-1	-1	2	0
N	-5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 5. The new scoring matrix derived from the dot product of the hypercomplex number representation of DNA bases.

which change the residues in a sequence, and insertions and deletions, which add or remove residues. Insertions and deletions are referred to as gaps. The total score assigned to an alignment is a sum of terms for each aligned pair of residues, plus terms for each gap.

An algorithm for finding an optimal alignment for a pair of sequences using an additive scoring system and gap penalties is called dynamic programming. Such algorithms are central to computational sequence analysis and are guaranteed to find the optimal scoring alignment. Better alignments have higher scores. Thus scores are maximized to find the optimal alignment.

A new scoring system is introduced by initially taking the dot product of the DNA base hypercomplex number representation shown in Table 1, $X \cdot Y = (P_{\mathrm{A}}^X, P_{\mathrm{T}}^X, P_{\mathrm{G}}^X, P_{\mathrm{C}}^X) \cdot (P_{\mathrm{A}}^Y, P_{\mathrm{T}}^Y, P_{\mathrm{G}}^Y, P_{\mathrm{C}}^Y) = P_{\mathrm{A}}^X P_{\mathrm{A}}^Y + P_{\mathrm{T}}^X P_{\mathrm{T}}^Y + P_{\mathrm{G}}^X P_{\mathrm{G}}^Y + P_{\mathrm{C}}^X P_{\mathrm{C}}^Y$. The new score values are then calculated using $s(X, Y) = X \cdot Y \times 20 - 5$, where the highest aligned score is 15 and the lowest one is -5, with a gap penalty of $d = 8$ for computational efficiency. After scaling the dot product value and rounding off to the nearest integer, the new scoring matrix is as shown in Fig. 5.

The conventional alignment algorithms (Durbin *et al*., 1998) are used together with the hypercomplex number representation of the base pairs and the new scoring model introduced here. A pair of DNA sequences is used throughout the rest of this paper as a demonstration of the feasibility of using this new scoring model:

H T A G A W M H R Y
T A W H C A M B H R

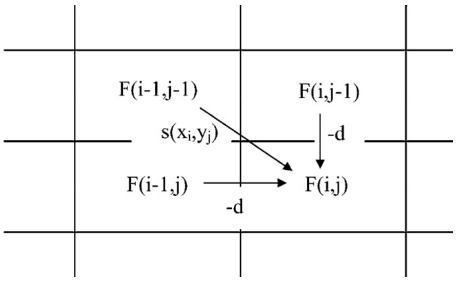

Figure 6. Derivation options for the $F(i, j)$ value.

5. GLOBAL ALIGNMENT USING HYPERCOMPLEX NUMBER REPRESENTATION

A matrix F indexed by i and j, one index for each sequence, is constructed, where the value of $F(i, j)$ is the score of the best alignment between the initial segment $X_1, X_2, \ldots, X_i$ and the initial segment $Y_1, Y_2, \ldots, Y_j$.

$F(i, j)$ are calculated with the knowns $F(i-1, j-1)$, $F(i-1, j)$, $F(i, j-1)$. The best score of an alignment is obtained in three ways: alignment of X_i with Y_j; or alignment of X_i with a gap; or alignment of Y_j with a gap. The best score up to (i, j) giving the optimal alignment is the highest of these three options. Hence,

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(X_i, Y_j), \\ F(i-1, j) - d, \\ F(i, j-1) - d. \end{cases}$$

The matrix of $F(i, j)$ values is built recursively by initializing $F(0, 0) = 0$, then filling the matrix from top left to bottom right using the other three values, as illustrated in Fig. 6. For boundary conditions along the top row where $j = 0$ and the leftmost column where $i = 0$, the values of $F(i, 0)$ and $F(0, j)$ are defined as $F(i, 0) = -id$ and $F(0, j) = -jd$. As the $F(i, j)$ value is filled, a pointer is kept in each cell back to the cell from which the value is derived.

The value in the final cell of the matrix is by definition the best score for an alignment of X and Y, which is the score of the best global alignment of X to Y. A traceback is done to find this global alignment by building the alignment in reverse, starting from the final cell and following the pointers kept when building the matrix. A pair of symbols is added onto the front of the current alignment with each step moved in the traceback process: X_i and Y_j if the step was to $(i-1, j-1)$, or X_i and the gap character '-' if the step was to $(i-1, j)$, or '-' and Y_j if the step was to $(i, j-1)$. This traceback procedure finds only one alignment with the optimal score. Thus an arbitrary choice is made between the two options if the derivations at any point are equal.

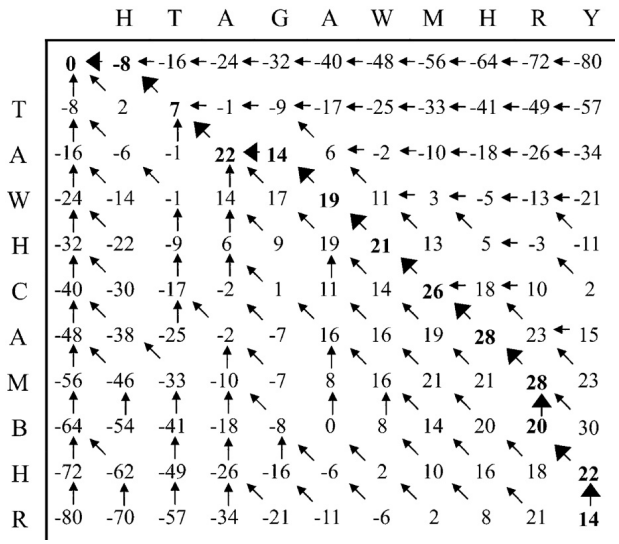

Figure 7. The global dynamic programming matrix for hypercomplex number representation of DNA sequences.

Because the score is a sum over independent pieces, this algorithm is feasible. This best score up to some point in the alignment is the best score up to the point one step before, plus the increment score of the new step.

Using the new scoring matrix in Fig. 7, the following global dynamic programming matrix is set up using the example DNA sequence pair.

From the above matrix, the corresponding optimal alignment of the two sequences with a total score of 14 is obtained as follows:

H	T	A	G	A	W	M	H	R	–	Y	–
–	T	A	–	W	H	C	A	M	B	H	R

5.1. ***Local alignment using hypercomplex number representation.*** Compared to the case of a global alignment, a more common situation occurs when the best alignment between subsequences of X and Y is required. An example of such is a comparison between extended sections of genomic DNA sequences. This alignment most sensitively detects similarity between two highly diverged sequences that might have a common evolutionary origin along their entire length. The highest scoring alignment of such subsequences is the best local alignment.

The difference lies in the feature that for local alignment, an extra possible value for $F(i, j)$ is added such that if all other options have a value of less than 0, $F(i, j)$ takes the value of 0:

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

		H	T	A	G	A	W	M	H	R	Y
	0	**0**	0	0	0	0	0	0	0	0	0
T	0	2	**15**	7	0	0	5	0	2	0	5
A	0	2	7	**30**	**22**	15	7	10	2	7	0
W	0	2	7	22	25	**27**	20	12	12	4	7
H	0	2	4	14	17	27	**29**	22	14	10	6
C	0	2	0	6	9	19	22	**34**	26	18	15
A	0	2	0	15	7	24	24	27	**36**	31	23
M	0	2	0	7	10	16	24	29	29	**36**	31
B	0	0	4	0	9	8	16	22	28	28	**38**
H	0	2	2	6	1	11	10	18	24	26	30
R	0	0	0	7	11	6	11	10	16	29	22

Figure 8. The local dynamic programming matrix for hypercomplex number representation of DNA sequences.

Once the value of $F(i, j)$ takes the option value of 0, a new alignment is started. The new option of 0 results in the top row and leftmost column taking the value of 0 instead of $-id$ and $-jd$ as in the case of global alignment.

In addition to the first difference, now in local alignment, an alignment could end anywhere in the matrix. The best score need not be in the bottom right corner. Instead, the traceback starts at the highest value of $F(i, j)$ over the whole matrix and ends when it reaches a cell with value 0 which corresponds to the start of the alignment.

The basis for this local alignment algorithm working is that the expected score for a random match must be negative, otherwise the scores for long matches between entirely unrelated sequences will be high on the basis of their lengths. As a result, the maximal scoring alignments would be global or nearly global although the algorithm is local. Similarly, there must be some score values higher then 0; if not, the algorithm cannot find any alignment at all.

Using the same pair of DNA sequences with hypercomplex number representation, the local dynamic programming algorithm is implemented to give the matrix in Fig. 8.

In the local dynamic programming matrix, it is not necessary to start the alignment at the bottom right cell. Instead, the alignment starts at the cell with the highest score so that the optimal local alignment can be found. In this case, the highest score is 38. Thus the traceback starts from there and ends when it reaches a score of 0. The optimal local alignment of this pair of example sequences has a score of 38 and is found to be

T A G A W M H R Y
T A – W H C A M B

5.2. ***Repeated matches using hypercomplex number representation.*** The best single local match between two sequences is easy to locate when the sequences are short. However, if one or both of them are long, it is probable that one will find many different local alignments with a significant score. None of these alignments should not neglected, as they are all evidence of a relation between the sequences. An example of such presence of many local alignments is provided by the many copies of repeated domains in a sequence.

Since there are always short local alignments with small positive scores even between entirely unrelated sequences, it is assumed that only matches with scoring higher than a threshold score, T, are considered.

Letting Y be the sequence containing the domain and X the sequence in which multiple matches are looked for, the same matrix is used as a demonstration, but the recurrence is now different. The value of $F(i, j)$ is derived differently. In the final alignment, X is separated into regions that match parts of Y in gapped alignments, and regions that are unmatched. The score of the completed match region is its standard gapped alignment score minus the threshold score, T. These match scores are positive. $F(i, j)$ for $j \geq 1$ is the best sum of match scores to $(X_1, X_2, \ldots, X_i)$, assuming that X_i is in a matched region, and the corresponding match ends in X_i and Y_j. Then, for the assumption that X_i is in an unmatched region, $F(i, 0)$ is the best sum of completed match scores to the subsequence $(X_1, X_2, \ldots, X_i)$.

As usual, $F(i, j)$ is initialized as $F(0, 0) = 0$. The matrix is then filled using the following recurrence relations:

$$F(i, 0) = \max \begin{cases} F(i-1, 0), \\ F(i-1, j) - T \end{cases} \qquad \text{where } j = 1, 2, \ldots, m,$$

and

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

The $F(i, 0)$ value is carefully derived to handle unmatched regions and ends of matches, allowing matches to end only when they have a score of at least T. The $F(i, j)$ value handles starts of matches and extensions. The total score has T subtracted for each match. When there are no matches of score greater than T, the total score is 0, as obtained by the repeated application of the $F(i-1, 0)$ option in the value of $F(i, 0)$.

The individual match alignments are then obtained by tracing back from cell $(n, 0)$ to (0,0), following the pointers kept. This traceback procedure is a global

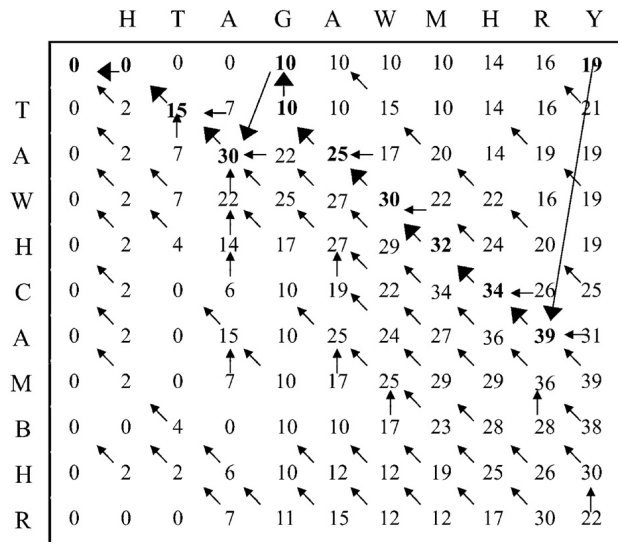

Figure 9. The repeat dynamic programming matrix for hypercomplex number representation of DNA sequences with threshold scores of $T = 20$.

procedure showing which residue in sequence Y is aligned with each residue in sequence X. The resultant global alignment contains sections of more conventional gapped global alignments of subsequences of X with subsequences of Y.

Likewise, by applying the algorithm for repeated matches with the new scoring model to the example sequences, the same DNA sequences demonstrate the outcome shown in Fig. 9.

For a threshold value of 20, the optimal alignment is

H	T	A	G	A	W	M	H	R	Y
–	T	A	T	A	W	H	C	A	•

When the threshold value is increased significantly, a large portion of the sequence is excluded from the matched region. In other words, a larger threshold score implies a higher stringency.

5.3. ***Overlap matches using hypercomplex number representation.*** Occasions arise when one sequence contains the other, or they overlap. This occurs often when fragments of genomic DNA sequences are compared to each other, or to longer chromosomal sequences. Thus another algorithm for such searches is required.

The algorithm for overlap matches is similar to that of global alignment, except that overhanging ends are not penalized. Hence the matching sequence starts on the top or left border of the matrix and ends on the right or bottom border.

The initialization is $F(0, 0) = 0$. The recurrence relations within the matrix are the same as those for global alignment. The highest score of the matching

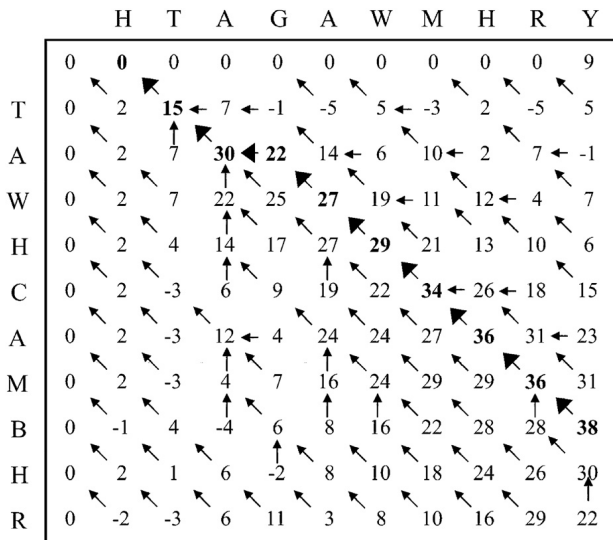

Figure 10. The overlap dynamic programming matrix for hypercomplex number representation DNA sequences with threshold of 20.

sequence is set on the right border (n, j) where $j = 1, 2, \ldots, m$, and the bottom border (i, m) where $i = 1, 2, \ldots, n$. The traceback starts from the point with the highest score and ends at the top or left edge of the matrix. Hence the governing algorithms for overlap matches are

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

and

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(X_i, Y_j), \\ F(i-1, j) - d, \\ F(i, j-1) - d. \end{cases}$$

The recursion for $F(i, 0)$ here is concerned only with the complete matches to $(Y_1, Y_2, \ldots, Y_m)$ instead of all possible subsequences of Y.

To find out whether the example hypercomplex DNA sequences show traces of overlapping in Fig. 10, they are subjected to the same overlapping dynamic programming. A threshold of 20 is pre-specified.

The possible overlap matching sequence is shown below. The optimal overlapping alignment has a score of 38. The resultant alignment is the same as that obtained for local alignment in the earlier section but this is not always true for other sequences.

T	A	G	A	W	M	H	R	Y
T	A	–	W	H	C	A	M	B

6. CONCLUDING REMARKS

To represent fully the DNA base-nucleic acid codes in hypercomplex numbers, a four-dimensional space is required. The representation number assigned to each base code takes the probability of each nucleotide in the DNA code into consideration. The conditions assumed in the assignment of the representation are that the probabilities for the occurrences of A, T, G and C are equal and the sum of the individual probabilities is 1.

The implementation of hypercomplex numbers in the dot matrix method brings forth an improvement to the conventional method (Mount, 2001) of placing a dot when there is a match between the corresponding residues of two sequences. As the hypercomplex number representation of DNA base-nucleic acid codes is in numbers instead of alphabetical characters, the significance of probabilistic sequencing is emphasized. To determine whether a dot should be placed between the aligned residues, the dot product of the hypercomplex number representation of the bases is taken and truncated. With the introduction of 'value' instead of 'dots' as in the conventional method (Mount, 2001), the truncation value can be varied and hence a greater control over the degree of alignment desired, besides the current control of window size and stringency, is possible. A higher truncation value corresponds to a higher stringency for longer matching regions between the sequences. With the addition of a new factor contributing to the outcome of the dot matrix diagram, more combinations of the three parameters can be selected to more aptly produce a more accurate dot matrix analysis for the desired condition of matches.

In addition, an implied advantage of the variable truncation value using the hypercomplex representation is that the sequences may not need to be further analyzed for actual matching regions using dynamic programming. The method of imaging may be used to overlap dot matrices of a similar pair of sequences but of increasing truncation value. As the truncation value increases, the number of dots is reduced. When the new matrix of higher truncation value is imposed on the previous matrix, a clearer picture of the location of the actual matching regions is superimposed on the screen.

To use the hypercomplex number representation of DNA sequences, a new scoring model has been derived. The new model, with the consideration of probability of each nucleotide presented in the DNA base-nucleic acid codes, uses the dot product arithmetic of the residues of the sequences to be matched. The dot product value is scaled and rounded off to an integer. The various algorithms have been applied to the sample sequence in hypercomplex number representation and the feasibility of using the hypercomplex number representation and scoring model has been verified. As most of the DNA codes consist of mixed bases, the alignments obtained for the various algorithms are very high. This is because the algorithms can detect a possible alignment with small possibility of a match between the two sequences.

REFERENCES

Durbin, R., S. R. Eddy, A. Krogh and G. Mitchison (1998). *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*, Cambridge: Cambridge University Press.

Lim, C. W. and J. -J. Shu (2001). On DNA modelling: interaction of a double helicoidal structure with viscous bio-fluid. *Automedica* **20**, 297–312.

Lim, C. W. and J. -J. Shu (2002). Studies on a DNA double helicoidal structure immersed in viscous bio-fluid, in: *ICCN 2002, Proceedings of the Second International Conference on Computational Nanoscience and Nanotechnology*, San Juan Marriott Resort and Stellaris Casino, San Juan, Puerto Rico, pp. 387–390.

Mount, D. W. (2001). *Bioinformatics: Sequence and Genome Analysis*, New York: Cold Spring Harbour Laboratory Press.

Needleman, S. B. and C. D. Wunsch (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* **48**, 443–453.

Smith, T. F. and M. S. Waterman (1981). Comparison of bio-sequences. *Adv. Appl. Math.* **2**, 482–489.

Received 29 October 2003 and accepted 23 January 2004