Bioinformatics Project (Programing with Matlab)

Implementation of Needleman-Wunsch algorithm with Matlab

1. - Why aligning sequences?

In molecular biology, we usually handle sequences both from DNA and proteins. These sequences contain information of thousands of years of evolution. Understanding the basic bionformatic tools to manage this information, at sequence level, is very important. Using those tools we can answer questions about what and when certain events happened and how do proteins work.

For this project, we will simplify the problem so it can be tractable in a reasonable and easy way.

Let consider the evolution concept as a simple change along time. The nucleotide sequences (and therefore the protein sequences) change during the evolutionary history.

Those changes, of random origin to simplify, usually do not confer an advantage to its carrier. In few occasions, it is possible that a DNA mutation does not affect the individual or it confers a slight disadvantage. If that is the case, a mutation would have some chance to spread in a population.

The molecular clock theory predicts that neutral changes at sequence level accumulate at a constant rate along the evolution. These neutral changes can explain the fact that evolutionary close species share more similar sequences (for example, man and orangutan) than species that diverged a long time ago.

When we have similar sequences, we call them homologues, that is, sequences that evolved from a common ancestor which in a certain moment diverged yielding the two species which are being compared.

The more different (or far, from the evolutionary/temporal point of view), more time passed from the divergence of the “ancestral” sequence.

To compare two sequences it is very important to measure quantitatively the sequence similarity and to establish a threshold to classify two candidates as homologues. These considerations are far from the scope of this project, and we will only focus in the alignment process to compare the sequences.

2. Sequence alignment is not a trivial problem

If we simplify our model, mutations can arise two possibilities:

- Element i from sequence s is equal to element j from sequence j: $s(i=1) == t(j=1)$
- Element i from sequence s is not equal to element j from sequence j: $s(i=1) \sim= t(j=1)$

However, these model is rather simplistic and is not taking account the possibility of delection (some parts of the sequence have been lost) or insertions (some parts have been inserted from different sources). If we include these problems, we have now three options:

- Align equivalent residues
- Align different residues
- Align residue with gap

Let's see and example of these possible cases. Let be s and t DNA sequence defined by:

s=ACGC

t=ACGA

Although it appears to be a simple problem, it is not. There are lot of ways to align these two sequences.

Some examples:

ACGC	ACGC-	-ACG-C	AC--GC	A-C-G-C
ACGA	ACG-A	A--CGA	ACGA--	-A-C-GA

Note that there is no gap‑gap alignment since it provides no information respect to time. Which alignment is the correct one? There is no such thing that a correct alignment. Since we are trying to reproduce what happened to the sequences during its evolution, we represent the biological information. From this point of view, the former examples give very different information: in the first alignment it we can see a mutation from C to A; in the second case, we see a deletion of A and a insertion of C, etc.

Try to decide which is the most biological relevant alignment is extremely complicate, so lets focus in obtaining all possible alignments. With all these aligments available, we have to score the different possibilities in a “biological way” (complex topic far away from the scope of this exercise).

It is well known that transitions (mutations of nucleotides of the same kind) are more frequent than transversions. To remind you the very basic of DNA chemistry, adenine (A) and guanine (G) are purines, while citosine (C) and timine (T) are pyrimidines. Then, transitions are mutations A/G, G/A, C/T and T/C, while transversions are the other possibilities: C/A, A/C, C/G, G/C, T/A, A/T, T/G, G/T.

If we include these problems, we can define a score which can consider very positive (+2) the fact that two sequence elements are equal, positive (+1) is there is a transitions, and negative (-1) if there is a transversion. Also to account the possibility of gaps, we add -2 to the a gap-element alignment.

$$(1)\ S = \begin{array}{c|cccc} & A & C & G & T \\ A & +2 & -1 & +1 & -1 \\ C & -1 & +2 & -1 & +1 \\ G & +1 & -1 & +2 & -1 \\ T & -1 & +1 & -1 & +2 \end{array}$$

$$(2)\ g = -2$$

Now, lets score the examples:

	ACGC ACGA	ACGC- ACG-A	-ACG-C A--CGA	AC--GC ACGA--	A-C-G-C -A-C-GA
Score	2+2+2-1= **+5**	2+2+2-2-2= +2	-2-2-2-1-2-1= -10	2-2-2-1-2-1= -6	-2-2-2-2-2-2-1= -12

3.- Needleman-Wunsch

Now that we know how to generate and score alignments, lets focus on obtaining an optimal alignment between two sequences from all the possibilities.

The aim of this project is implement the Needleman-Wunsch algorithm with the score matrix 1 and to find the optimal alignment of two sequences.

This algorithm was proposed by Saul Needleman y Christian Wunsch en 1970 [1], to solve the problem of optimization and storage of maximum scores. It is not a very complicated algorithm so lets see step-by-step how it works.

First, we create a matrix D, with size (m+1)x(n+1), where m and n are the lengths of the sequences t and s, where we are going to store the score of alignment sub-problems. We look for a global alignment, that is, alignments of sequences s and t as a whole. The best possible score for an alignment of s and t is always at D(m,n), the last values of the matrix. Each value D(i,j) is the maximum score of a fragment s(1...i) and the fragment t(1...j). In each position of the matrix D, we will store the maximum score for all possible local alignments to i,j positions, so we do not have to recalculate these scores again.

$$(3)\ D(i,j) = \max\begin{Bmatrix} D(i-1,j-1)+S(i,j) \\ D(i-1,j)+g \\ D(i,j-1)+g \end{Bmatrix}$$

If we have an optimal alignment from D(i-1,j-1), we just have three options for the next position i,j

1. Sequence elements s(i) and t(j) are the same
2. We insert a gap in t
3. We insert a gap in s

Note that we have added one gap at the beginning of each the sequence (that explain the matrix size (m+1)x(n+1)). The main reason for these gaps is the absence of previous elements needed for the elements marked with “?”

Using the parameters depicted above (1) and (2) and the formula (3) we can fill the matrix D.

Hint: Initialize the matrix with zeros and use a different loop to initialize the first column and the first row.

4.- Trace back

Once the matrix is filled, we just have to trace back the path from the D(m,n) position to the D(0,0) position. To do so, we have to re-calculate values for D(i-1, j-1), D(i-1, j) and D(i,j-1) to recall where was the maximum values corresponding to D(i,j).

1. If we came from D(i-1, j-1), sequence elements i and j are aligned.
2. If we came from D(i-1, j), there is a gap in sequence s
3. If we came from D(i, j -1), there is a gap in sequence t

More than an optimal path is also possible, but in this project we only ask for one of them.

		i=0	*i*=1	*i*=2	*i*=3	*i*=*m*
		-	A	C	G	C
j=0	-	0	?			
j=1	A	?	?			

5. ¿Which is the aim of my program?

First, your function should ask for two vectors (sequences s and t):

$s(i), t(i) \in X$
$X = \{A, C, G, T\}$

and return as output two vectors and one scalar value. The vector should have sequences s' and t'

$s'(i), t'(i) \in X'$
$X' = \{A, C, G, T, '-'\}$

and the scalar, should have the score for the optimal aligment.

6.- I'm lost or stuck, what should i do?

Ok, the problem is complicated, but it is not so complicated that you cannot be able to solve it.

If you understand the algorithm (which is part of the so-call dynamic programming) you have more than half way done. If you experience problems please send a e-mail to jklett@cbm.uam.es and acortes@cbm.uam.es

[1] Needleman SB, Wunsch CD. (1970). “A general method applicable to the search for similarities in the amino acid sequence of two proteins". *J Mol Biol* **48** (3): 443–53