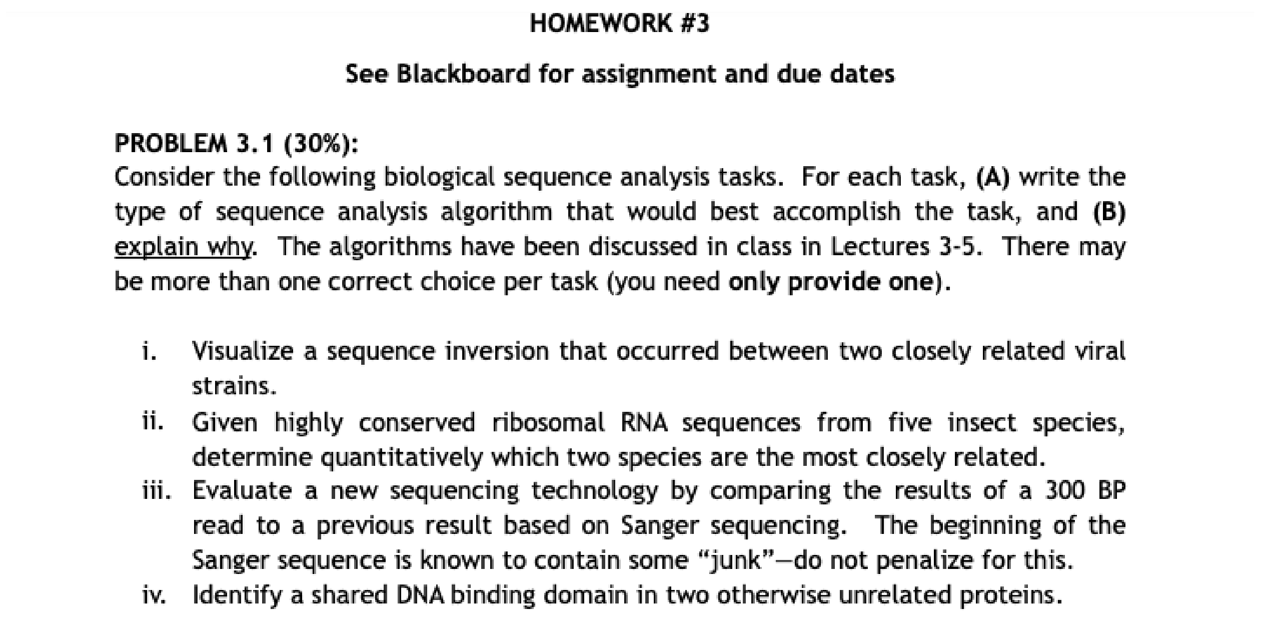


HOMEWORK #3

See Blackboard for assignment and due dates

PROBLEM 3.1 (30%):
Consider the following biological sequence analysis tasks. For each task, **(A)** write the type of sequence analysis algorithm that would best accomplish the task, and **(B)** explain why. The algorithms have been discussed in class in Lectures 3-5. There may be more than one correct choice per task (you need **only provide one**).

i. Visualize a sequence inversion that occurred between two closely related viral strains.
ii. Given highly conserved ribosomal RNA sequences from five insect species, determine quantitatively which two species are the most closely related.
iii. Evaluate a new sequencing technology by comparing the results of a 300 BP read to a previous result based on Sanger sequencing. The beginning of the Sanger sequence is known to contain some "junk"—do not penalize for this.
iv. Identify a shared DNA binding domain in two otherwise unrelated proteins.

i. Dot plots. Given that they are very closely related strains and the requirement is to visualize the inversion, the use of dot plots is an option that can provide a qualitative and directly visible similarity analysis.

ii.Global sequence alignment (Needleman- Wunsch dynamic programming algorithm). Because the sequences are highly conserved ribosomal RNA sequences and we need to have a quantitative result of the best alignment, global sequence alignment can satisfy this, producing the optimal global alignment of two sequences, including gaps.

iii. Semi-Global Alignment. Because the requirement is do not penalize for the meaningless beginning of Sanger sequencing, and semi-global can help to ignore the contribution of one noisy end of the sequence and still give us a global alignment result for evaluating the new sequence technology.

iv. Local sequence alignment
This is because the two proteins are not related except for a common structural domain, which is determined by matching sequence substrings. And by using local sequence matching we can align that conserved segment of two very dissimilar sequences.

PROBLEM 3.2 (30%):

(A) Manually fill out a dynamic programming (DP) matrix for the pair of short DNA sequences below using the Needleman-Wunsch global alignment algorithm, and the scoring scheme below

(B) Include traceback arrows

(C) Report the score of the optimal alignment(s)

(D) Write out one possible optimal alignment

(E) How many equally optimal alignments are there (i.e., are there any ties)?

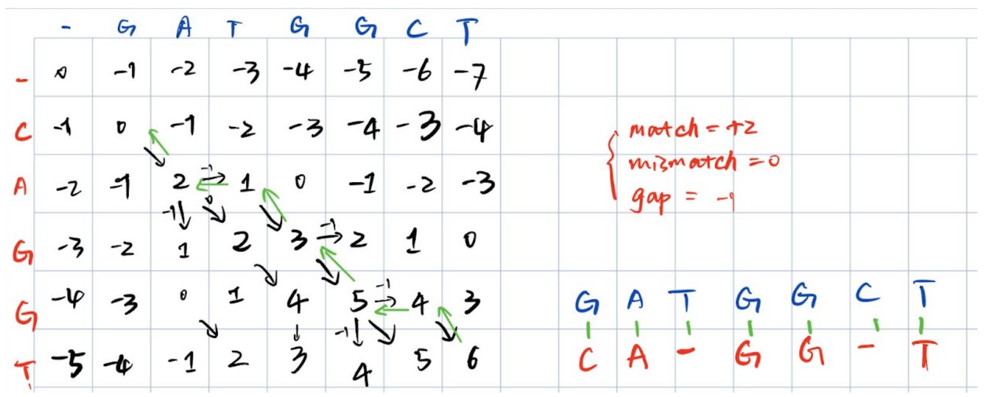
Scoring scheme: match = +2; mismatch = 0; gap = -1

Sequence 1 = `GATGGCT`

Sequence 2 = `CAGGT`

Note: You may opt to use/fill in the matrix template provided on Blackboard (also shown below) for this problem.

(A)(B)(D)



	-	G	A	T	G	G	C	T
-	0	-1	-2	-3	-4	-5	-6	-7
C	-1	0	-1	-2	-3	-4	-3	-4
A	-2	-1	2	1	0	-1	-2	-3
G	-3	-2	1	2	3	2	1	0
G	-4	-3	0	1	4	5	4	3
T	-5	-4	-1	2	3	4	5	6

match = +2
mismatch = 0
gap = -1

```
G A T G G C T
| | | | | | |
C A - G G - T
```

(C) score = 6

(E) No, there is only one optimal alignment.

PROBLEM 3.3 (40%):

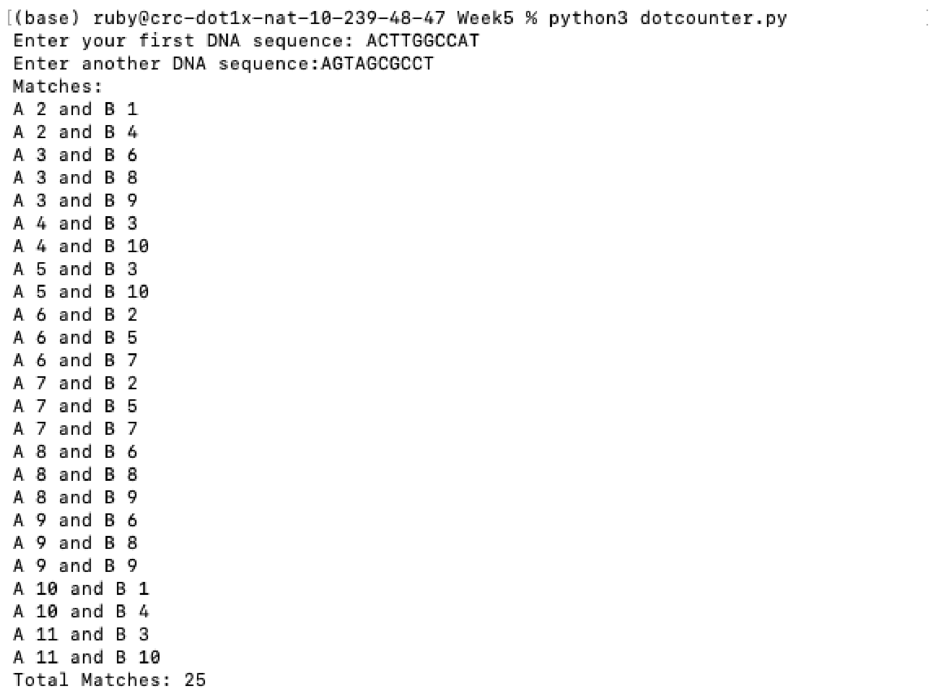
(A) Write a python script called `dotcounter.py` that, given a pair of strings A and B, determines the total number of dots that would appear in the dot plot comparing the two strings and prints their coordinates. Your program does *not* need to draw the actual dot plot! For example, if the two strings are A = **GGGAC** and B = **CGGAC**, then your program would output:

```
Matches:
A1 and B2
A1 and B3
A2 and B2
...
A5 and B5
Total matches:
9
```

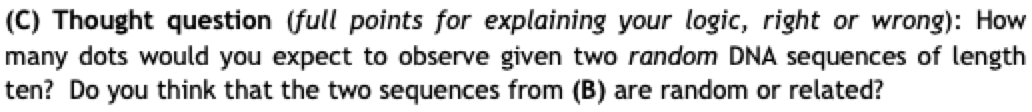
Refer to the dot plot below to check that this is correct.

	C	G	G	A	C
G		●	●		
G		●	●		
G		●	●		
A				●	
C	●				●

(B) Run your program with the two input strings A = **ACTTGGCCAT** and B = **AGTAGCGCCT** and include the results (list of matches and number of total matches) in your homework.



```
[(base) ruby@crc-dot1x-nat-10-239-48-47 Week5 % python3 dotcounter.py
 Enter your first DNA sequence: ACTTGGCCAT
 Enter another DNA sequence:AGTAGCGCCT
 Matches:
 A 2 and B 1
 A 2 and B 4
 A 3 and B 6
 A 3 and B 8
 A 3 and B 9
 A 4 and B 3
 A 4 and B 10
 A 5 and B 3
 A 5 and B 10
 A 6 and B 2
 A 6 and B 5
 A 6 and B 7
 A 7 and B 2
 A 7 and B 5
 A 7 and B 7
 A 8 and B 6
 A 8 and B 8
 A 8 and B 9
 A 9 and B 6
 A 9 and B 8
 A 9 and B 9
 A 10 and B 1
 A 10 and B 4
 A 11 and B 3
 A 11 and B 10
 Total Matches: 25
```



(C) Thought question (*full points for explaining your logic, right or wrong*): How many dots would you expect to observe given two *random* DNA sequences of length ten? Do you think that the two sequences from **(B)** are random or related?

To align two random DNA sequences of length 10, there would be a dot plot with 10 multiplied 10 locations.

For each location(i,j) (i<10, j< 10) in the plot, the possibility of a dot (match) =$4 \times \frac{1}{4} \times \frac{1}{4} =$ ¼,

and a blank(mismatch) =$1 - \frac{1}{4} =$ ¾,

$$\begin{cases} P_{dot} = \frac{1}{4} \\ P_{blank} = \frac{3}{4} \end{cases}$$

the possibility of total n dots would be a Bernoulli event with success probability = $P_{dot} = 1/4$,

the probability of n dots (P_n) :

$$P_n = C_{100}^n \times (P_{dot})^n \times (P_{blank})^{(100-n)} = C_{100}^n \times (\frac{1}{4})^n \times (\frac{3}{4})^{(100-n)} = C_{100}^n \times (\frac{1}{4})^{100} \times 3^{(100-n)}$$

$$P_n = \frac{100!}{n!(100-n)!} \times (\frac{1}{4})^{100} \times 3^{(100-n)}$$

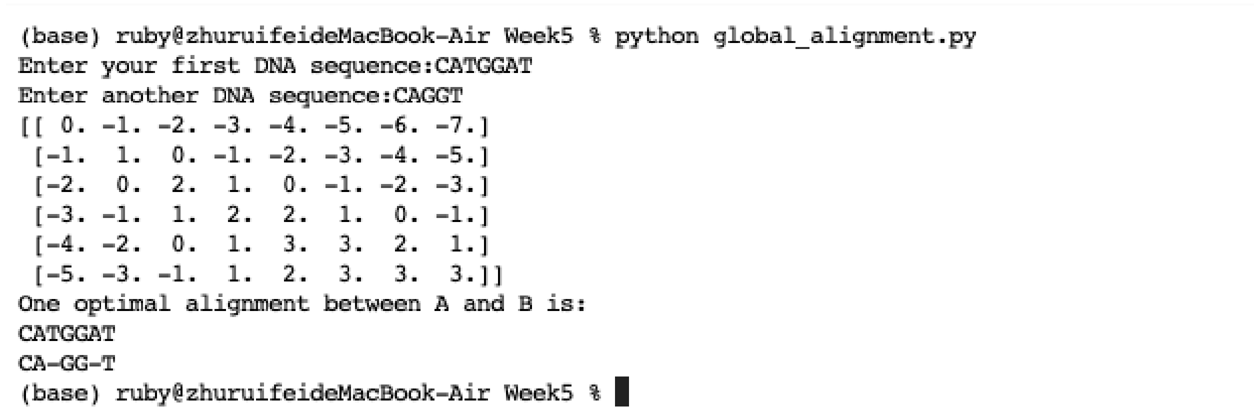
The highest probability is located at n= 25.

So the two sequences in (B) are likely to be random.

PROBLEM 3.4 Extra Credit (20%):

Write a python script called `global_alignment.py` that would extend `dotcounter.py` to incorporate Needleman-Wunsch algorithm to do a global alignment between two sequences.
Suggested scoring matrix: match=1, mismatch =0, gap = -1.

One output:



```
(base) ruby@zhuruifeideMacBook-Air Week5 % python global_alignment.py
Enter your first DNA sequence:CATGGAT
Enter another DNA sequence:CAGGT
[[ 0. -1. -2. -3. -4. -5. -6. -7.]
 [-1.  1.  0. -1. -2. -3. -4. -5.]
 [-2.  0.  2.  1.  0. -1. -2. -3.]
 [-3. -1.  1.  2.  2.  1.  0. -1.]
 [-4. -2.  0.  1.  3.  3.  2.  1.]
 [-5. -3. -1.  1.  2.  3.  3.  3.]]
One optimal alignment between A and B is:
CATGGAT
CA-GG-T
(base) ruby@zhuruifeideMacBook-Air Week5 % 
```